Anais do XII SEMINCO

Seminário de Computação
5 a 8 de Agosto de 2003
Centro de Convenções Willy Sievert - PROEB

Promoção

Universidade Regional de Blumenau - FURB
Pró-Reitoria de Extensão e Relações Comunitárias - PROERC
Centro de Ciências Exatas e Naturais - CCEN
Departamento de Sistemas e Computação - DSC
Centro Acadêmico Livre de Computação - CALCOMP

Comissão Organizadora

Prof. Carlos Eduardo Negrão Bizzotto (Presidente)
Prof. Wilson Pedro Carli (Coordenador)
Prof. Alexander Roberto Valdameri
Prof. Antônio Carlos Tavares
Prof. Everaldo Artur Grahl
Prof. Jomi Fred Hübner
Profa. Joyce Martins
Prof. Ricardo Alencar de Azambuja
Prof. Roberto Heinzle

Ficha Catalográfica Elaborada pela Biblioteca da FURB

Seminário de Computação (12.: 2003 : Blumenau, SC)

S471a Anais do XII SEMINCO / promoção Universidade Regional de Blumenau, Departamento de Sistemas e Computação; Wilson Pedro Carli (coordenador). - Blumenau, O Departamento, 2003. 220 p. : il.

1. Computação - Congressos. I. Carli, Wilson Pedro. II. Universidade Regional de Blumenau. Departamento de Sistemas e Computação. III. Título.

CDD 004

Universidade Regional de Blumenau

Reitor
Prof. Egon José Schramm
Vice-Reitor
Prof. Rui Rizzo
Diretor do Centro de Ciências Exatas e Naturais
Prof. Sérgio Stringari
Chefe do Departamento de Sistemas e Computação
Prof. Wilson Pedro Carli
Coordenador do Colegiado do Curso de Ciências da Computação
Prof. Everaldo Artur Grahl
Coordenador do Colegiado do Curso de Sistemas de Informação
Prof. Roberto Heinzle

Apresentação

A Universidade Regional de Blumenau - FURB, através do Departamento de Sistemas e Computação realiza o XII SEMINCO - Seminário de Computação no mês de agosto de 2003, fazendo parte do evento GeNeTec 2003 - Congresso de Gestão, Negócios e Tecnologias junto com a Feira Blusoft Brasil 2003. A consolidação do SEMINCO como evento científico nas áreas de Computação e Informática fez com que a organização do GeNeTec também ficasse sob responsabilidade da Universidade. Isto demonstra que a idéia de um evento interno voltada puramente para acadêmicos do curso de Ciências da Computação iniciada a 12 anos, hoje atende uma comunidade de informática que envolve os acadêmicos dos cursos de Ciências da Computação e Sistemas de Informação, profissionais da área de informática e empresários.

Desta forma, agradecemos a todos os envolvidos na organização dos dois eventos, bem como a Comissão de Avaliação Inter Institucional que não mediu esforços em avaliar os diversos artigos submetidos conforme à chamada de trabalhos. Destes artigos,originários de diversas Instituições de Ensino Superior do Brasil, 17 foram selecionados para constar nestes anais, cuja organização é feita conforme as seguintes áreas de conhecimento: Computação Gráfica, Engenharia de Software, Informática na Educação, Inteligência Artificial, Integração Hardware / Software, Sistemas de Informação e Redes de Computadores.

Esperamos que nos quatro dias de realização dos dois eventos atender as expectativas dos participantes e formulamos um novo convite para a participação na próxima edição.

Comissão Organizadora

Agradecimentos

Conselho Nacional de Pesquisa - CNPQ
Comissão de Avaliação Inter Institucional
Empresas Patrocinadoras
Fundação de Ciência e Tecnologia de Santa Catarina - FUNCITEC
Núcleo de Informática - FURB
Projeto Acredito - FURB
Sociedade Brasileira de Computação - SBC

Comissão de Avaliação Interinstitucional

Adelmo Luis Cechin (UNISINOS - RS)
Alexander Roberto Valdameri (FURB - SC)
Ana Lúcia Anacleto Reis (FURB - SC)
Anita da Rocha Fernandes (UNIVALI - SC)
Anna Helena Reali Costa (USP - SP)
Antonio Carlos Tavares (FURB - SC)
Dalton Solano dos Reis (FURB - SC)
Edson Satoshi Gomi (USP - SP)
Eliseo Berni Reategui (UCS - RS)
Everaldo Artur Grahl (FURB - SC)
Fabiane Barreto Vavassori (UNIVALI - SC)
Fernando Santos Osório (UNISINOS - RS)
Francisco Adell Péricas (FURB - SC)
Graça Marietto (USP - SP)
Gustavo G. Lugo (USP - SP)
Heitor Strogulski (UCS - RS)
Iraci Cristina da Silveira (UCS - RS)
Jaime Simão Sichman (USP - SP)
Jomi Fred Hübner (FURB)
José Leomar Todesco (INE/CTC/UFSC - SC)
Joyce Martins (FURB - SC)
Marcelo Thiry (UNIVALI - SC)
Marcos Eduardo Casa (UCS - RS)
Mauro Marcelo Mattos (FURB - SC)
Maurício Capobianco Lopes (FURB - SC)
Miguel Alexandre Wisintainer (FURB - SC)
Paulo Cesar Rodacki Gomes (FURB - SC)
Rafael Cancian (UNIVALI - SC)
Rafael Heitor Bordini (Univ. of Liverpool)
Reinaldo Bianchi (USP - SP)
Renata Vieira (UNISINOS - RS)
Roberto Heinzle (FURB - SC)
Rogerio Golçalvez Bittencourt (UNIVALI - SC)
Sérgio Stringari (FURB - SC)
Tiaraju Asmuz Diverio (UFRGS - RS)

Artigos Selecionados

Computação Gráfica

Uma Nova Classificação e um Protocolo de Concepção de Comunidades Virtuais 9

Adriano Gheller Bruschi (ESALQ/USP) e Daniel Weller (UNIMEP)

Uma Arquitetura Java para Resolução de Grafos Estendidos de Restrições em Prototipação Virtual Distribuída 23

Paulo César Rodacki Gomes (FURB), Joyce Martins (FURB), Jaderson Trierweiler (FURB) e Daniel Moser Trallamazza (FURB)

Engenharia de Software

Avaliação da Interface do Auto-Atendimento das Agências do BESC: uma Abordagem Ergonômica 33

Vera Schuhmacher Niedersberg Schumacher (UNISUL), Maria Inés Cstiñeira (UNISUL), Aidê Luzia A. de Souza (UNISUL) e Roselane M. M. Camargo (UNISUL)

Análise comparativa de algoritmos de grafos para um Sistema de Auxílio à Matrícula de Alunos 45

Rogerio Sorroche (FURB) e Mauricio Capobianco Lopes (FURB)

Protótipo de um ambiente de monitoramento e apresentação de programas Java utilizando Reflexão Computacional 57

Marcel Hugo (FURB) e Romeu Gadotti (FURB)

GRAMO: Uma Técnica para a Construção de Modelos de Domínio Reutilizáveis no Desenvolvimento de Sistemas Multiagente 71

Carla Gomes de Faria (UFMA) e Rosario Girardi (UFMA)

Padrões baseados em Agentes para a Modelagem de Usuários 85

Ismênia Ribeiro (UFMA) e Rosario Girardi (UFMA)

Uso de Design Patterns e J2EE: um estudo de caso 99

Rogerio Sorroche (FURB) e Mauricio Capobianco Lopes (FURB)

Informática na Educação

Ferramentas que auxiliam o desenvolvimento da lógica de programação 113

Rafael de Santiago (UNIVALI) e Rudimar Luís Scaranto Dazzi (UNIVALI)

Inteligência Artificial

Agentes Coletores na Gerência de Redes de Computadores 121

Eduardo Erle dos Santos (UFSC), Fernando Luiz Koch (UFSC), Marcos Dias Assuncao (UFSC) e Carlos Becker Westphal (UFSC)

Ferramenta de Pré e Pós-processamento para Data Mining 131

Deborah Ribeiro Carvalho (IPARDES / UTP), Marcos Bueno (IPARDES / UTP), Wilson Alves Neto (IPARDES /UTP) e Luiz Ricardo Lopes (UTP)

Integração Hardware/Software

Computador de bordo automotivo baseado em PC-Linux 141

Cristiano Freese (FURB) e Miguel Alexandre Wisintainer (FURB)

Avaliação e Adequabilidade de Benchmark Paralelos Modernos 155

Omar Andres Carmona Cortes (ICMC-USP), Regina Helena Carlucci Santana (ICMC-USP) e Marcos José Santana (ICMC-USP)

Sistemas de Informação

Web Services: estudo de caso envolvendo uma aplicação B2B 169

Cristiano Fornari Colpani (FURB) e Alexander Roberto Valdameri (FURB)

Utilização da Pesquisa Tabu na geração de um Sistema de Informação Geográfica aplicado ao problema de localização de torres de rádio transmisão 179

Leandro Toss Hoffmann (UNISINOS) e Arthur Tórgo Gómez (UNISINOS)

Rede de Computadores

Aplicação que usa Protocolo de Perfil Leve para Transferência de Arquivos 193

Cleber Machado Ortiz (UFRGS), Lisandro Zambenedetti Granville (UFRGS) e Liane Margarida Rockenbach Tarouco (UFRGS)

Modelagem e Análise de Desempenho de uma Rede Baseada em Tecnologia MPLS 207

Aujor Tadeu Cavalca Andrade (UFSC) e Carlos Becker Westphall (UFSC)

Uma Nova Classificação e um Protocolo de Concepção de Comunidades Virtuais

Adriano Gheller Bruschi (ESALQ/USP)
driko@tequila.esalq.usp.br

Daniel Weller (FCMNTI - FEAU / UNIMEP)
dweller@uol.com.br

Resumo. O objetivo deste trabalho é apresentar resultados de um estudo sobre as comunidades virtuais, com a identificação dos elementos que potencializam as comunidades como instrumento de agregação para o entretenimento. Uma breve conceituação sobre o tema e uma proposta de classificação amplamente utilizada são apresentadas. O trabalho apresenta principais resultados de análises quantitativas e qualitativas de diversas comunidades para entretenimento digital. A partir dos resultados é apresentada uma nova abordagem de categorização: invariante do tipo, dos temas/assunto considerados e das ferramentas disponíveis, instrumentos do entretenimento digital.

Palavras-chave: Comunidades virtuais, Entretenimento digital, Interatividade

1 Introdução

Com o surgimento e o desenvolvimento das Novas Tecnologias de Comunicação e Informação, novas formas de relacionamento são propiciadas. Hoje, a Internet oferece um universo de possibilidades que está revolucionando a maneira de interagirmos. O conceito de uma comunidade sem fronteiras, digital, está se desenvolvendo rapidamente para assumir o lugar da comunidade geocêntrica tradicional [Beach 1997].

O conceito de comunidades é costumeiramente usado para descrever um conjunto de pessoas em uma determinada área geográfica, a qual pode possuir uma estrutura social, com relacionamento entre os indivíduos. Além disso, também pode existir um espírito compartilhado entre os membros da comunidade e um sentimento de pertencer ao grupo [Lévy 1996].

A Internet propiciou o maior fenômeno social, revolucionando a comunicação humana e criando novas formas de convívio humano, utilizando os meios computacionais [Dizard 2000, Lévy 1999]. A experiência que nós vivemos hoje é única. Pessoas nos mais diferentes lugares interagindo em tempo real e reunidas em busca de interesses comuns, através dos mais variados serviços, se reúnem diariamente em todas as partes do mundo.

A rede é uma ferramenta tecnológica poderosa que pode ser utilizada para inclusão, permitindo promover o desenvolvimento das comunidades por favorecer justamente seu elemento fundamental: a interação humana, os relacionamentos, sem considerar os aspectos geográficos com a quebra da barreira do espaço, com a superação dos limites temporais de sicronicidade e de classe sociais - limitantes nos relacionamentos reais [Pearce 1997]. Uma comunidade virtual é um grupo de pessoas que se comunicam por uma rede de computadores distribuídos. Algumas comunidades são formais, com regras e deveres, critérios de admissão e até mensalidades pagas pelos membros. Outras são agrupamentos menos formais com fronteiras imprecisas e com um quadro de associados flutuante [Powers 1997].

É importante destacar que existem diversos aspectos que colaboram para que um sítio construa um sentimento de pertencer a um grupo virtual, a uma comunidade afim, onde os participantes se transformam em "habitantes" desse espaço comunitário. Muito esforço é realizado de forma a atingir esse objetivo. O caminho para constituição de uma comunidade virtual, passa

pelo fornecimento de informações personalizadas e contextualizadas, disponibilização de serviços/funcionalidades computacionais e outros elementos motivacionais para participação, manutenção de relacionamentos, construção de conhecimento e inclusão no grupo virtual.

2 Tipos de Comunidades

No mundo virtual, assim como em nosso cotidiano, existe uma alta diversidade de comunidades. Powers [Powers 1997] propôs uma categorização muito utilizada, a qual considera a existência de três tipos principais, a saber: o MUD, a sala de bate-papo e o espaço 3D. Segue abaixo uma descrição de cada uma dessas formas.

2.1 MUD

MUD, abreviação para *Multi-User Dimension* segundo os promotores de seus aspectos sociais ou *Multi-User Dungeon* pelos que recordam que *MUDs* se desenvolveram dos famosos jogos de aventura da década de 1970, é um mundo inteiramente descrito por parágrafos de texto. Nele, a interação ocorre por meio de comandos, que geram resultados que podem ser vistos e respondidos pelos outros participantes.

Os *MUDs* e seus predecessores são umas das mais antigas formas de comunidades virtuais desenvolvidas para redes de computadores. São as formas menos visuais, que utilizam o poder da escrita para capturar a imaginação, o que em momento algum é um problema para a criação de uma comunidade virtual. Pelo contrário, os *MUDs* ganham com isso um grande significativo social quando comparados às comunidades gráficas. Os participantes podem achar mais fácil descrever idéias elaboradas e ações expressivas por meio da escrita.

A dependência entre as palavras escritas e a imaginação dos participantes pode ser observada na formação de um senso de lugar nos *MUDs*. A cada vez que se entra em uma sala, um parágrafo descreve o local, as saídas, e quem está presente no momento. Os habitantes conversam e representam um papel em cada sala. Eles conversam diretamente ou indiretamente descrevendo suas próprias ações, que permite aos participantes dar conotações especiais ao texto de seus personagens.

As conexões sociais entre os habitantes podem tornar-se muito complexas ao decorrer do tempo de envolvimento. Os habitantes possuem um sentimento muito grande de posse de seu mundo *MUD*. São muito protetores do local e dos outros habitantes. Como um gerenciador de um *MUD*, é possível perceber uma lealdade única assim como expectativas incomparáveis no que fazer para o melhor da comunidade. Não são recomendadas modificações irregulares e arbitrárias nas descrições do mundo ou freqüentes mudanças no software. É imperativo que um ambiente *MUD* permaneça estável, pois esta forma de comunidade é muito sensível a transtornos e quedas de performance.

Um *MUD* irá comportar uma comunidade de média escala, variando de 50 até 300 habitantes em determinado momento.

2.2 Chats

As salas de bate-papo ou *chats* são um dos mais populares serviços da Internet atualmente. Dos três tipos de comunidades virtuais, a sala de bate-papo é o tipo que melhor suporta conversações a respeito de tópicos variados. As salas de bate-papo dependem dos participantes para gerar conversações curtas de determinados tópicos. Estas comunidades não suportam conversas longas, pois não é mantido um histórico. A sala é um conjunto de participantes que se reúnem para conversar a respeito de tópicos de seus interesses.

Geralmente um participante inicia em uma sala comum, uma "sala de estar", onde os participantes se reúnem antes de decidirem seguir para salas dedicadas a um tópico específico. Uma vez que um grupo se forma ou um participante decide entrar em algum tópico, o mesmo ou o grupo se movem para uma sala dedicada exclusivamente a este tópico. O tamanho da sala geralmente é limitado para permitir conversas menos fragmentadas.

As salas de bate-papo são ideais para as conversas casuais. Essa tecnologia não é tão adequada quanto os *MUDs* no desenvolvimento de comunidades duradouras, pois as pessoas que visitam as salas de bate-papo geralmente se movem de tópico em tópico ao invés de ambiente para ambiente. Caso a sala esteja vazia num determinado momento, não será uma experiência adequada aos os novos visitantes. Alguns operadores de salas de bate-papo, os quais podem ser contratados ou voluntários, podem até montar equipes para constantemente estimularem conversas nas salas.

2.3 Espaços 3D

Uma interface de computador que propicia as mesmas imagens, sons e até mesmo as sensações da vida real. Esse é o conceito da Realidade Virtual (RV), um ambiente fictício com o qual o usuário pode se relacionar de maneira intuitiva, exatamente como se relacionaria com o mundo real. Todo o processamento ocorre em computadores dotados de dispositivos de entrada e saída, que permitem o usuário interagir de maneira mais humana e menos mecânica com o computador. A interação pode ocorrer também através de uma aplicação de RV bem desenhada, que irá convidar o usuário a lidar com o ambiente renderizado como se fosse a "vida real" fora do computador. Além da interface amigável, a interatividade é outra característica da Realidade Virtual. O usuário não se comporta como um mero espectador passivo, mas sim como um participante ativo, podendo até mesmo ser o centro de atenção dos acontecimentos. O objetivo dessa interação pode ser a educação, o entretenimento ou a telepresença [Promise 1995].

A telepresença permite que o usuário projete uma personalidade imaginária, podendo criar uma representação de si próprio no Ciberespaço de outra pessoa e vice-versa, interagindo com ela através dessa representação. No Ciberespaço compartilhado o usuário e seus companheiros podem manipular o ambiente virtual.

Nas comunidades virtuais, a forma mais popular pela qual ela se revela é através dos espaços 3D, que é a mais nova tecnologia. Quando se trata do aspecto visual, estão no topo em relação aos *MUDs* e bate-papos. Nos espaços 3D os participantes são capazes de vasculhar mundos tridimensionais simulados enquanto se comunicam, embora a comunicação ainda não seja o ponto forte desta tecnologia. Os participantes são atraídos pelo desejo de explorar grandes áreas ou resolver enigmas baseados em pontos localizados por todo um espaço.

Um espaço 3D é uma localidade virtual compartilhada criada por um sistema de computador. A possibilidade de exploração dos participantes se dá como numa exploração de um prédio. Eles podem abrir portas, subir rampas e até usarem elevadores e tele-transportes, tudo renderizado por computadores [Powers 1997].

Todo o participante é representado no espaço 3D por um avatar, que é uma personagem virtual assumida pelos participantes, incluindo representação gráfica de um modelo estrutural de corpo, modelo de movimento, modelo físico e outras características, como roupas e acessórios [Weller 2000]. O avatar não precisa necessariamente representar uma figura de um ser humano, pode também ser um animal, alienígena, uma planta etc. Os participantes vêem os outros andando e interagindo com o ambiente e podem até falar uns com os outros como numa sala de bate-papo. As comunidades formadas nos espaços 3D são transitórias, e o maior foco é o visual.

A linguagem *VRML* foi criada em 1995 por Dave Ragget da *Hewlett-Packard* européia. Atualmente a versão *VRML97*, criada em 1997, é uma linguagem standard para objetos

tridimensionais, cenas ou mundos veiculados pela Internet, que permite também a interação e manipulação dos objetos ou cenas, assim como o exame deles por todos os pontos de vista. Com relação aos mundos de *VRML*, estes podem ser animados e pode ser permitido para o usuário criar um avatar para ativamente participar e interagir com outros habitantes [Prado 2000].

Atualmente, são vários os desenvolvedores e criadores que utilizam a linguagem *VRML* para implementar espaços 3D. Esse fato é explicado especialmente pelos fatos de os *plug-ins* para visualização e o acesso dos visitantes serem fáceis e gratuitos.

3 Desenvolvimento

A forma como as comunidades virtuais se estabelecem e oferecem níveis de interação a seus usuários mudaram e evoluíram bastante. Muitas podem ser as justificativas, como por exemplo: a explosão de uso da Internet, o desenvolvimento tecnológico das ferramentas, o aumento do número e da diversidade dos usuários da Internet, a procura em atender a demanda dos interesses dos internautas e a convergência das mídias e o entretenimento digital [Forman 2000].

Com essas mudanças foi identificada a necessidade de estabelecimento de uma nova classificação, mais adequada para as mudanças e evoluções ocorridas. Nessa nova classificação o processo de categorização é considerado a partir da constatação de que as comunidades virtuais atualmente se organizam em um molde ou *template* principal comum, o qual é composto por ferramentas de interação e integração com funcionalidades semelhantes. As diferenciações aparecem no nível superior: a partir dos interesses, temas e conteúdos que serão oferecidos aos usuários.

Essa nova abordagem classificatória foi resultado da realização de análises quantitativas e qualitativas de diversas comunidades virtuais para entretenimento digital, todas com forte carga motivacional para a inclusão digital. Os resultados obtidos reforçam a abordagem de categorização por molde: com o núcleo baseado nas ferramentas disponíveis, facilitadoras da inclusão digital, e invariante ao tipo, aos temas/assuntos e conteúdos considerados.

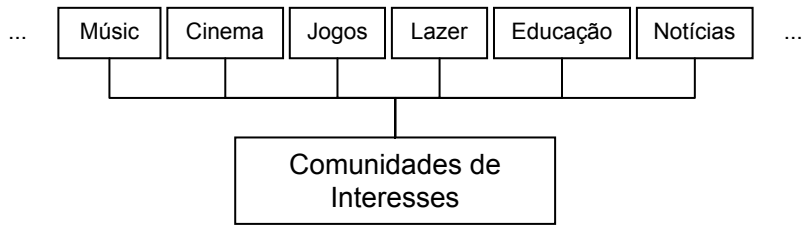


Figura 1: Molde principal comum e diferenciações a partir dos temas

Para verificar a validade dessa abordagem foram analisadas algumas comunidades que oferecem interações com e entre os usuários e que possam ser separadas em diferentes categorias de assunto e temas de interesse [Bruschi 2001]. Considerando que o entretenimento digital corresponde a um dos fatores motivacionais para superar a barreira do analfabetismo digital, auxiliando na inclusão social, foram selecionadas quatro categorias de assunto e algumas comunidades representativas. A saber: **Cinema e Artes:** Cinema em Cena, Cineclick, Cinema na Sala, Heróis; **Jogos:** Outer Space, Jogos.com.br, Ação Games, GamePower; **Humor:** Humor Tadela, Central de Piadas, Piadas Web Page; **Música:** Guns N'Roses Brasil, Demo Club, Usina do Som, Whiplash!.

3.1 Elaboração do Questionário

Baseado nas análises dos *websites* selecionados foi elaborado um questionário, cujo objetivo é identificar os hábitos dos usuários numa comunidade e em suas principais ferramentas de interatividade, como bate-papo, listas de discussão e fóruns. O questionário foi distribuído pessoalmente para 25 pessoas identificadas como usuárias da Internet e das ferramentas de interatividade propostas. Esta população foi composta de alunos de Ciência da Computação e funcionários de duas empresas relacionadas a desenvolvimento de programas. No processo do preenchimento do questionário os usuários foram auxiliados e direcionados a não deixarem nenhuma das questões em branco. Entretanto, como dois (2) usuários não tinham familiaridade nenhuma com algumas ferramentas propostas (um com fórum e um com listas de discussão), 23 questionários foram respondidos completamente e 2 foram respondidos parcialmente. Além desta distribuição em mãos, o questionário também foi disponibilizado via Internet e anunciado para diversos usuários em potencial via e-mail. Não foi possível determinar quantas pessoas efetivamente receberam ou leram a chamada para o preenchimento do questionário. No entanto, 30 pessoas participaram preenchendo completamente todas as seções do questionário. Os resultados foram recebidos e tabulados juntamente com os dados dos questionários entregue em mãos.

4 Resultados e Discussão

Com os resultados das análises das comunidades selecionadas, foram elaborados diversos quadros de funcionalidades e ferramentas de interação para cada uma das comunidades consideradas [Bruschi 2001]. A proposta de abordagem desse trabalho é reforçada, a partir da análise dos resultados. Os quadros e suas considerações seguem abaixo:

Funcionalidades	Quantidade de comunidades que a funcionalidade foi encontrada	Situação em que a ferramenta é recomendada para implementação em uma comunidade
Bate-papo	9	Oferecer interação em tempo real entre os visitantes
Lista de discussão	8	Proporcionar o desenvolvimento de relacionamentos entre os usuários
Fórum	9	Idem a "Lista de discussão"
Enquetes	6	Forma de obter *feedback* a uma ação feita ou a ser feita
Comércio eletrônico	6	Comercializar produtos ou serviços e levantar fundos para a manutenção da comunidade
Jogos	6	Garantir a diversão do usuário. Dependendo do objetivo da comunidade, pode desviar o foco
Pesquisa de dados pessoais do usuário	1	Conhecer os usuários habitantes da comunidade
Livro de visitas	2	Oferecer oportunidade para críticas e sugestões
Cartões virtuais	4	Meio de propaganda. Dissemina a *URL* e nome da comunidade para possíveis novos habitantes
Exigência de login para acesso interno	5	Implementar quando disponibilizar conteúdo de interesse privado
Oportunidade de publicação de artigos do usuário	3	Caso objetive uma comunidade voltada à educação considera-se interessante
Contato com o(s) desenvolvedor(es)	15	Contato direto do usuário com o desenvolvedor

Figura 2: Quadro de funcionalidades x comunidades das localidades avaliadas

Este quadro e as observações dos serviços em comum mais utilizados vem reforçar a idéia inicial proposta neste trabalho, de que as comunidades dispõem atualmente de um mesmo modelo de ferramentas de interatividade, com destaque para o uso de bate-papos, listas de discussão e fóruns. As comunidades são diferenciadas apenas pelo conteúdo que oferecem ao usuário.

Com a observação dos resultados das ferramentas mais utilizadas, foi elaborado o questionário para avaliar a utilização destas ferramentas pelos usuários, analisando principalmente as formas como são oferecidas numa comunidade e também a preferência do usuário, de como deveriam ser aplicadas caso não concordem com a implementação da comunidade que participam. Com relação aos resultados, os mesmos foram tabulados propiciando quatro quadros, mostrados e discutidos abaixo.

Questões	Respostas				
O que o faz torná-lo visitante assíduo de um *site*?	Design bem-feito	Conteúdo interessante	Conteúdo atualizado	Interação com outros visitantes	Webmaster ativo
	26,2%	23,4%	36,9%	4,5%	9%
Você participa de pesquisa de dados pessoais num *site*?	Sim		Não		
	41,8%		58,2%		
Qual seu principal método para conhecer um site?	Imprensa	Buscadores	Links e/ou banners	Amigos	
	18,2%	54,5%	5,5%	21,8%	
Você leva em consideração os dados de visitas diárias ou totais?	Sim		Não		
	27,3%		72,7%		
Você mantém contato com o webmaster ou equipe de desenvolvimento?	Sim		Não		
	36,4%		63,6%		
Você costuma assinar e/ou ler o livro de visitas dos *sites*?	Sim, leio	Sim, assino	Sim, assino e leio	Não	
	11%	14,5%	0%	74,5%	
Você participa de enquetes de um *site*?	Sim		Não		
	81,9%		18,1%		

Figura 3: Quadro do comportamento dos usuários nas comunidades virtuais (amostra de 55 pessoas).

No quadro acima, buscou-se avaliar os principais motivos que levam o usuário a escolher um *website* e como esse usuário leva em consideração a qualidade do *website*. Também foi possível verificar se o usuário mantém interação através de algumas ferramentas, como livro de visitas, enquetes, pesquisas pessoais ou ainda se mantém contato com o desenvolvedor do *website* ou equipe de desenvolvimento para eventuais sugestões ou críticas.

É importante destacar que um resultado inesperado ocorreu, indicando que o fator "interação com outros usuários" não está sendo determinante para um usuário se tornar visitante assíduo. Os usuários preferem que o conteúdo oferecido seja o qual eles procuram e principalmente que esse conteúdo seja atualizado constantemente. Após os usuários consolidarem sua assiduidade ao *website*, eles tendem a desenvolver laços sociais com os outros participantes ou mesmo com o desenvolvedor do *website*, através das ferramentas de interação que são oferecidas.

Outra observação importante se faz com relação à ferramenta livro de visitas. A principal razão que um *webmaster* disponibiliza tal funcionalidade em sua comunidade, é oferecer um espaço para os usuários participarem, deixando seus elogios ou críticas ao *website* e seu autor. No entanto, esta ferramenta não vem sendo tão utilizada pelos usuários quanto havia se estimado antes dos resultados. Apenas 14,5% do total participante utilizam a ferramenta para deixar seus comentários, enquanto que 74,5% dos usuários não participam, seja lendo ou postando mensagens.

Ainda assim, se considera interessante e importante oferecer um espaço para os comentários dos visitantes. Como esta ação parte do interesse dos usuários em estarem contribuindo para o crescimento da comunidade onde habitam, não são muitos os que estão dispostos a estarem ativos com suas opiniões, sugestões e críticas, mas sim uma pequena parcela.

Questões	Respostas			
Você já usou que tipos de bate-papo?	Web	*IRC*	3D	Outros
	41,13%	44,9%	4,67%	9,35%
Que tipo mais lhe agrada e usa?	Web	*IRC*	3D	Outros
	25,45%	65,45%	5,45%	3,65%
Você participa com que freqüência em bate-papos?	Todos os dias	2 a 5 vezes por semana	1 vez por semana	raramente
	38,2%	18,2%	10,9%	32,7%
A sala que participa é moderada ou não?	Sim	Não	Não sei	
	58,2%	18,2%	23,6%	
Você prefere que uma sala seja monitorada (por operadores e/ou *bots*) ou livre?	Monitorada	Livre		
	52,7%	47,3%		
Já deixou de freqüentar um bate-papo por causa do administrador ou dos usuários?	Por causa do administrador	Por causa dos usuários	Não	
	25,4%	27,3%	47,3%	
Adicionaria em seu *site* (se tiver ou tivesse) que tipo de interface de bate-papo?	Web	*IRC*	3D	Outros
	49,1%	49,1%	1,8%	0%
Se oferecesse um bate-papo em seu *site*, o manteria monitorado ou livre?	Monitorado	Livre		
	61,8%	38,2%		

Figura 4: Quadro do comportamento dos usuários nos bate-papos (amostra de 55 pessoas).

Com os dados do quadro acima, verificou-se que a interface *IRC*, além de ser a mais usada, é a mais adequada e agradável aos usuários. É certo que não se trata de uma interface amigável e simples ao usuário iniciante. A popularidade do *IRC* aqui indicada se deve à sua praticidade para os usuários experientes, pois se trata de uma implementação destinada especificamente ao serviço de bate-papo, com uma performance mais eficaz e rápida quando comparada às interfaces *web*. É importante destacar também que no *IRC* é estabelecida uma interface padrão de comandos, diferentemente das interfaces *web*, onde existem inúmeras implementações com seus respectivos comandos e funcionalidades.

Ainda traçando um paralelo entre os usuários de interfaces *IRC* e *web*, nota-se uma diferença considerável quanto à freqüência de uso. Enquanto dos usuários de *IRC* 38,9% freqüentam o bate-papo todos os dias, os usuários de bate-papo via *web* totalizam 50%, fato explicado pela simplicidade da interface *web* aos novos usuários e pelo marketing agressivo de divulgação do sistema, como no caso de portais de conteúdo, com destaque ao Universo Online.

Um resultado interessante surgido com esta pesquisa foi que a moderação seja ela feita por operadores e/ou *bots* (programas robô), também influi na freqüência dos usuários. Enquanto que dos usuários que participam de bate-papos moderados 65,6% do total freqüentam todos os dias, dos usuários que participam de bate-papos livres nenhum usuário relatou freqüentar todo dia, ficando o dado mais próximo 20% do total freqüentando de duas a cinco vezes por semana. Tal fato pode ser explicado que com um determinado monitoramento do bate-papo, os usuários se sentem mais seguros quanto a problemas de atritos, baixo nível e propagandas (*spam*). Inclusive, os usuários relataram que no caso de oferecerem o serviço de *chat*, a escolha seria de um sistema moderado. No entanto, não seria recomendado logo de início implementar uma sala de bate-papo moderada.

Dessa forma, deve-se primeiramente oferecer o serviço de forma livre, conquistando um público que seja fiel e constante, caso contrário poderá ocorrer excessos de moderação e regulamentos desnecessários para um fluxo de usuários muito baixo, facilmente gerando insatisfações. Inclusive recomenda-se manter um monitoramento em um bate-papo com freqüência

igual ou superior a 50 usuários simultaneamente conectados. É importante para o administrador da comunidade ter a consciência de que com um sistema monitorado, a exigência de atenção e dedicação ao bate-papo aumentará, sendo necessário que faça o recrutamento de uma equipe para ajudar o monitoramento. O tamanho desta equipe irá variar de acordo com o fluxo de usuários. Outra opção caso o administrador não tenha dedicação total é a utilização de *bots* (programas-robô), geralmente implementados em sistemas *UNIX* ou *Linux* e que podem ser configurados para filtrarem todo o fluxo de mensagens da sala de bate-papo.

Questões	Respostas			
Com que freqüência participa lendo e/ou postando mensagens?	Todos os dias	2 a 5 vezes por semana	1 vez por semana	Raramente
	35,2%	16,7%	5,5%	42,6%
Você participa somente lendo ou também postando mensagens?	Somente leio		Leio e posto	
	50%		50%	
Você recebe as mensagens da lista de que forma?	Individual	Digest	Web	
	68,5%	0%	31,5%	
Você acha que é possível fazer amizades duradouras em listas?	Sim		Não	
	72,2%		27,8%	
Você conhece os comandos para assinar ou cancelar a lista?	Sim		Não	
	77,8%		22,2%	
Você já chegou a cancelar uma lista por causa da política do administrador ou dos usuários?	Por causa do administrador	Por causa dos usuários	Não	
	18,5%	16,7%	64,8%	
A lista que você freqüenta é moderada ou livre?	Moderada	Livre	Não sei	
	38,9%	44,44%	16,66%	
Você prefere que uma lista seja moderada ou livre?	Moderada		Livre	
	46,3%		53,7%	
Caso adicionasse ao seu *site* (se tiver ou tivesse) uma lista, seria moderada ou livre?	Moderada		Livre	
	53,7%		46,3%	

Figura 5: Quadro do comportamento dos usuários nas listas de discussão (amostra de 54 pessoas).

Com o quadro acima, algumas são as conclusões tiradas para o serviço de listas de discussão. O modo preferido do recebimento das mensagens é o individual, ou seja, a cada mensagem e-mail enviada à lista, o usuário recebe uma cópia em seu endereço eletrônico pessoal, dispensando um boletim diário reunindo todas as mensagens do dia (modo *digest*) ou ainda checarem via *web* no *website* da lista (modo *no-mail*). Quanto à moderação, apesar dos usuários participarem de listas livres e preferirem elas assim, no caso de virem a oferecer o serviço de lista em seus *websites* (se tiverem ou tivessem), o fariam de forma moderada.

Esta é uma decisão muito importante, pois além de tirar uma considerável parcela do senso de comunidade por não permitir os usuários tratarem de seus próprios problemas e atritos, irá exigir muita dedicação do *webmaster*, pois numa lista de porte médio (300 pessoas), a média mensal é de 400 a 600 mensagens. Como numa lista moderada os e-mails devem ser lidos pelo administrador e aprovados ou não, será recomendado optar pelo modo de postagem livre caso não exista a possibilidade do administrador estar acompanhando diariamente o fluxo da lista, pois caso isso ocorra, os usuários podem facilmente ficar insatisfeitos com o serviço.

É importante destacar que, apesar de impedir determinadas situações onde os usuários poderiam estar aptos a solucionar e exigir uma dedicação grande, a opção pela moderação auxilia na integração de crianças e pessoas que poderiam ter problemas com expressões de baixo nível ou mau comportamento de outros usuários participantes. Além disso, listas moderadas tendem a

receber mais acessos diários, pois 100% dos que assinam listas moderadas participam diariamente, enquanto que dos usuários de listas livres, apenas 25% acompanham todos os dias.

Questões	Respostas			
Com que freqüência você acessa o fórum?	Todos os dias	2 a 5 vezes por semana	1 vez por semana	Raramente
	20,3%	9,3%	3,7%	66,7%
Você participa somente lendo ou também postando mensagens?	Somente lendo		Lendo e postando	
	51,9%		48,1%	
O fórum exige registro para o acesso de postagem ou é livre?	Restrito		Livre	
	31,5%		68,5%	
Você prefere que o fórum tenha acesso restrito ou livre?	Restrito		Livre	
	27,8%		72,2%	
Você já deixou de participar de um fórum por causa da política do administrador ou dos usuários?	Por causa do administrador	Por causa dos usuários	Não	
	3,7%	5,55%	90,75%	
Caso adicionasse ao seu *site* (se tiver ou tivesse) um fórum, deixaria o acesso restrito ou livre?	Restrito		Livre	
	37,04%		62,96%	

Figura 6: Quadro do comportamento dos usuários nos fóruns (amostra de 54 pessoas).

Da análise deste quadro, pode-se concluir que apesar de inúmeras comunidades estarem implementando o serviço de fóruns, estes ainda não são tão usados como se esperava. A maior parcela dos usuários participa raramente e ainda assim somente lendo as mensagens, sem estarem efetivamente postando suas opiniões ou novos assuntos. Tal fato explica-se pela preferência às listas de discussão, onde as mensagens são distribuídas via e-mail, diferentemente dos fóruns, onde o usuário tem que estar on-line para ler e/ou postar. Como muitos usuários ainda não dispõem de conexão à Internet que não seja via linha telefônica, eles tendem a economizar seu tempo on-line.

Nos fóruns, os usuários preferem participar em fóruns de postagem livre e caso tenham ou tivessem uma comunidade e fossem fornecer o serviço, também disponibilizariam o serviço com a postagem livre. No entanto, realizando uma análise comparativa entre o tipo de postagem (restrita ou livre) com a freqüência de acesso, nota-se que com um fórum de acesso mediante uma identificação (*username* e *password*), os usuários tornam-se mais freqüentes, com 64,7% acessando todos os dias. Do total de participantes de fóruns livres, nenhum usuário relatou acessar todos os dias (2,7% acessa uma vez por semana e 97,3% do total acessa raramente).

5 Conclusão

Através da análise dos resultados do primeiro quadro (figura 2), juntamente com os resultados e discussões dos demais quadros elaborados a partir dos resultados do questionário, foi possível elaborar e concluir um protocolo de concepção para uma comunidade virtual de porte médio a partir de um *website* da *web*. Este protocolo visa proporcionar que o administrador de um *website* comum possa seguir determinados passos e instruções, para completa ou parcialmente aplicar ferramentas de interatividade com o objetivo de fazer com que seu *website* seja uma comunidade virtual de porte pequeno ou até mesmo porte médio. Segue abaixo o protocolo, constituído das ferramentas identificadas neste trabalho e os respectivos passos, sugestões e instruções para a implementação de cada uma delas.

Bate-papo: Serviço a ser aplicado quando se deseja oferecer um espaço para os usuários de forma síncrona trocarem mensagens e desenvolverem seus laços sociais. É uma ferramenta com um forte poder de integração e interação entre os usuários. Através da análise do questionário, observou-se que os usuários preferem e optariam pela interface *IRC* no caso de uma implementação. No entanto, deve-se lembrar que ela não é amigável a novos usuários e isso pode gerar insatisfação. Deve-se inicialmente avaliar e estudar o público alvo da comunidade e caso este ofereça a confiança para que se implemente uma sala em interface *IRC*, faz-se a escolha por essa tecnologia. Caso contrário, ou ainda se for de desejo de agregar um fluxo constante de novos e inexperientes usuários, deve-se optar por um bate-papo com interface *web*. É importante lembrar que a interface *IRC* permite o vínculo com um canal de bate-papo em servidores *IRC*, possibilitando a integração de um número maior de usuários. Esta característica não se encontra presente na interface *web*. Relacionado à escolha de manter o bate-papo livre ou monitorado por uma equipe de operadores, cabe também a análise dos usuários. Caso seja uma interface *IRC* e com um número diário de usuários igual ou maior que 50, deve-se optar pelo monitoramento. Se o número de usuários for superior a 100 pessoas por dia, recomenda-se ainda a implementação de um *bot* configurado para aplicar as punições determinadas no regulamento de uso do canal. No caso de um bate-papo com interface *web*, não se recomenda a moderação, pois se tratam geralmente de usuários iniciantes e com pouca intimidade ao uso do bate-papo e suas funcionalidades.

Lista de discussão: Ferramenta a ser disponibilizada quando existe a intenção de oferecer um espaço em que os usuários possam interagir em diversas discussões através do e-mail. Os comandos para assinar a lista e cancelar a assinatura, devem estar bem claros ao usuário. Também é aconselhável fornecer a possibilidade dos usuários receberem a lista em modo normal (individual), *digest* (resumo) ou ainda não receberem no e-mail e checarem via *web*, pois os usuários podem possuir hábitos e horários diferentes, e oferecendo diversas opções, se ganha um número maior de participantes. Com relação ao formato das mensagens da lista, recomenda-se que opte por fornecer as opções de escolha do usuário, entre as mensagens em formato texto e as mensagens em formato HTML, com o padrão sendo o formato texto, pois além de uma determinada faixa dos usuários podem não possuir um programa leitor de e-mails apto a trabalhar com o formato HTML, o modo *digest* de recebimento, geralmente transforma mensagens HTML em formato texto, comprometendo assim o formato e conteúdo inicial. A respeito da opção de aplicar a moderação, recomenda-se usar tal prática somente em listas com mais de 100 usuários e ainda assim se o moderador tiver disponibilidade diária de ler pelo menos de 30 a 50 e-mails, que é a média diária de uma lista de porte médio. Caso contrário, recomenda-se optar por deixar a postagem de mensagens livre, sendo as mensagens distribuídas diretamente.

Fórum: Ferramenta de funcionalidade semelhante às listas de discussão, mas diferenciada por ser desenvolvida em interface *web*. Pode-se optar por dois modos principais de aplicação: a discussão somente sobre tópicos determinados pelo administrador da comunidade ou liberar a postagem de novos tópicos pelos participantes. Neste quesito o administrador do *website* deve avaliar qual o escopo que a ferramenta deverá ter, assim determinando qual aplicação escolher. Pode ser aplicado um módulo de identificação antes do acesso de postagem, para garantir a participação somente de assinantes ou então deixar a postagem livre a qualquer participante, sem a necessidade de um registro. É importante observar que, com base nos dados do quadro 5, nos fóruns onde existia um sistema de autenticação, os usuários tornaram-se mais freqüentes.

Enquetes: Deve-se dispor deste recurso quando se deseja a opinião dos usuários sobre um determinado tópico, que pode ser inclusive, a respeito de alguma funcionalidade da própria comunidade. É uma ferramenta poderosa, especialmente se permitir que juntamente com a opção escolhida, o usuário possa fazer alguns comentários, sugestões ou críticas.

Classificados: Serviço para proporcionar transações comerciais entre os usuários. Através de uma interface *web*, os usuários podem postar anúncios de vendas, compras ou trocas, ou ainda contatar via e-mail algum anunciante de um produto que tenham se interessado. Recomenda-se aplicar um módulo de identificação ou então minimamente manter um registro com os endereços IP e respectivos horários de acesso dos usuários que postem anúncios, de forma a manter a credibilidade do sistema.

Cartões virtuais: Ferramenta que proporciona que o usuário envie (remetente) a um conhecido (destinatário) um cartão virtual. O destinatário é avisado por e-mail acessa a comunidade, e com seu código acessa o cartão. Especialmente usado em datas comemorativas, tem um apelo interessante que é justamente no caso de o destinatário não ser um usuário que conheça a comunidade. Nesse caso, existe a possibilidade de que a comunidade ganhe um usuário freqüente.

Indique o *website*: Seu uso é permitir um usuário indicar o endereço da comunidade a um conhecido, para que esse a visite e possivelmente, se torne um usuário freqüente.

Jogos: Para proporcionar um passatempo e diversão aos usuários, podem ser implementados jogos via *web*, com destaque a jogos que promovam a competição entre os usuários, como concursos e adivinhações. Para estimular a participação, recomenda-se oferecer prêmios aos mais bem colocados.

Livro de visitas: Apesar de identificado no trabalho como um serviço pouco utilizado, o uso dos livros de visita faz-se recomendado, pois é um espaço no qual o usuário deixa sua apreciação pessoal a respeito do *website*, geralmente com elogios ou críticas construtivas. Como o usuário não tem a tendência a entrar diretamente em contato com o *webmaster* ou a equipe de desenvolvimento via e-mail, este espaço cumpriria tal funcionalidade.

Estatísticas de acesso: Através de estatísticas como contadores de acessos totais, acessos diários e mensais, o usuário pode acompanhar o desenvolvimento do *website* e seu fluxo de usuários. Em particular, uma implementação interessante é a de permite exibir quantos usuários estão acessando a comunidade simultaneamente. É importante destacar que para medir o sucesso de uma comunidade, não se faz somente o uso do número de visitas, mas também do tempo gasto pelo usuário nas ferramentas e funcionalidades oferecidas pela comunidade.

Das instruções acima e da avaliação das observações realizadas, é possível que qualquer administrador de um *website* de porte médio venha a desenvolver uma comunidade virtual, oferecendo diversas ferramentas de integração e interação.

Convém destacar que, além das estratégias e ferramentas de interatividade, como alavanca e instrumento de inclusão digital, a comunidade deve oferecer conteúdo de qualidade e de interesse específico para os membros da comunidade e realizar manutenções constantes, de forma a motivar o acesso e o retorno dos usuários.

Referências

BEACH, David. **Transformando públicos em comunidades**. p. 301-340. In Szeto, Butterick, McKirchy-Spencer e outros. “Interatividade na Web: Transforme seu site em uma experiência inesquecível”. Berkeley Brasil Editora. 1997.

BRUSCHI, Adriano G. e Weller, Daniel. **Comunidades Virtuais: Uma Proposta de Classificação e Um Protocolo de Concepção**. UNIMEP. 2001.

DIZARD, W. **A Nova Mídia**. Jorge Zahar Editor. 2000.

FORMAN, P. and Saint John, R. W. **Creating Convergence**. p. 34-40. In: The Future of Digital Entertainment. Scientific American. 2000.

LÉVY, P. **O que é o Virtual?**. Editora 34. 1996.

LÉVY, P. Cibercultura. Editora 34. 1999.

PEARCE, C. **The Interactive Book**. Macmillan Techincal Publishing. 1997.

POWERS, Michael. **How to program a virtual community**. Ziff-Davis Press. 1997.

PRADO, Gilbertto. **Desertesejo**. In Cadernos da Pós-Graduação, Instituto de Artes, Unicamp, vol 4 nº 1, 2000, pp. 40-53.

PROMISE, Valerie e Van Buren, Christopher. **Realidade Virtual**. p. 107-124. In Harris, Stuart e outros. "CYBERLIFE!". Berkeley Brasil Editora. 1995.

WELLER, D. **Ciberdrama**. In: Da Criação ao Roteiro. Editado por Doc Comparato. Editora Rocco. 2000.

Uma Arquitetura Java para Resolução de Grafos Estendidos de Restrições em Prototipação Virtual Distribuída

Paulo César Rodacki Gomes (FURB)
rodacki@inf.furb.br

Joyce Martins (FURB)
joyce@inf.furb.br

Jaderson Trierweiler (FURB)
trier@inf.furb.br

Daniel Mozer Trallamazza (FURB)
tylenol@inf.furb.br

Resumo. A prototipação virtual torna-se muito mais interessante quando sólidos 3D são objetos dinâmicos distribuídos em uma rede de computadores. Entretanto, a natureza dinâmica do processo de design requer uma investigação mais profunda sobre objetos reativos 3D. O presente artigo usa o conceito de Grafos Estendidos de Restrições com reatividade bidirecional para propor uma arquitetura para prototipação virtual distribuída baseada na linguagem Java, no CORBA e na especificação de uma nova linguagem chamada ECGL. A viabilidade da proposta é apresentada através da implementação inicial de uma aplicação simples .

Palavras-chave: Prototipação Virtual, CAD distribuído, Grafos Estendidos de Restrições, CORBA, Java.

1 Introdução

A prototipação virtual (FEIJÓ, 2001) é o processo de construção de um artefato virtual completo, também chamado de maquete eletrônica, de forma que problemas de projeto (design) e manufatura possam ser antecipados e discutidos em um ambiente de trabalho colaborativo, sustentado por uma ferramenta de projeto auxiliado por computador (CAD - Computer Aided Design). A prototipação virtual requer uma arquitetura de sistema de CAD na qual os objetos que compõem o artefato virtual, notadamente objetos 3D, estejam interligados em um ambiente de trabalho colaborativo. E mais, estes objetos podem estar localizados em computadores distribuídos em uma rede. Neste tipo de ambiente, chamado ambiente virtual para o trabalho colaborativo em sistemas de CAD, a atividade de projeto do artefato possui um caráter altamente dinâmico, pois alterações ocorridas em um objeto 3D podem acarretar em alterações em outros objetos 3D.

As relações entre os objetos 3D formam um sistema de restrições. Esse sistema pode ser modelado através de um grafo especial, chamado de Grafo Estendido de Restrições (ECG), proposto por Gomes (1999).

O presente trabalho apresenta uma proposta para implementação de ECGs baseada numa arquitetura cliente-servidor, em Java. O comportamento dinâmico dos ECGs é tratado através da definição de uma gramática para definição dos grafos e da implementação, usando a linguagem Java, do respectivo interpretador.

A seção 2 deste artigo apresenta alguns aspectos relacionados ao problema de Prototipação Virtual Distribuída, em especial, os ECGs. Na seção 3, é apresentada uma proposta de arquitetura para tratamento de ECGs baseada em CORBA e na API Java 3D. Na arquitetura proposta, o

tratamento de ECGs é suportado por uma Linguagem de Especificação de ECGs, apresentada na seção 4. Finalmente, as discussões e conclusões são apresentadas na seção 5.

2 Prototipação virtual distribuída

A figura 1 ilustra a complexidade de um ambiente para prototipação virtual, onde um artefato completo encontra-se disperso em uma rede composta de diferentes plataformas, sistemas operacionais, equipamentos de visualização e equipes de projetistas. Neste ambiente integrado, os processos altamente dinâmicos são suportados por barramentos de geometria e tecnologia de objetos distribuídos. O barramento de geometria possibilita a interoperabilidade e permite que os projetistas cooperem entre si usando diferentes softwares de CAD na rede. O ACIS (SPATIAL TECHNOLOGY, 2003) e o CORBA (OBJECT MANAGEMENT GROUP, 1995) têm sido propostos respectivamente como barramento de geometria e arquitetura de objetos distribuídos para sistemas de CAD integrados.

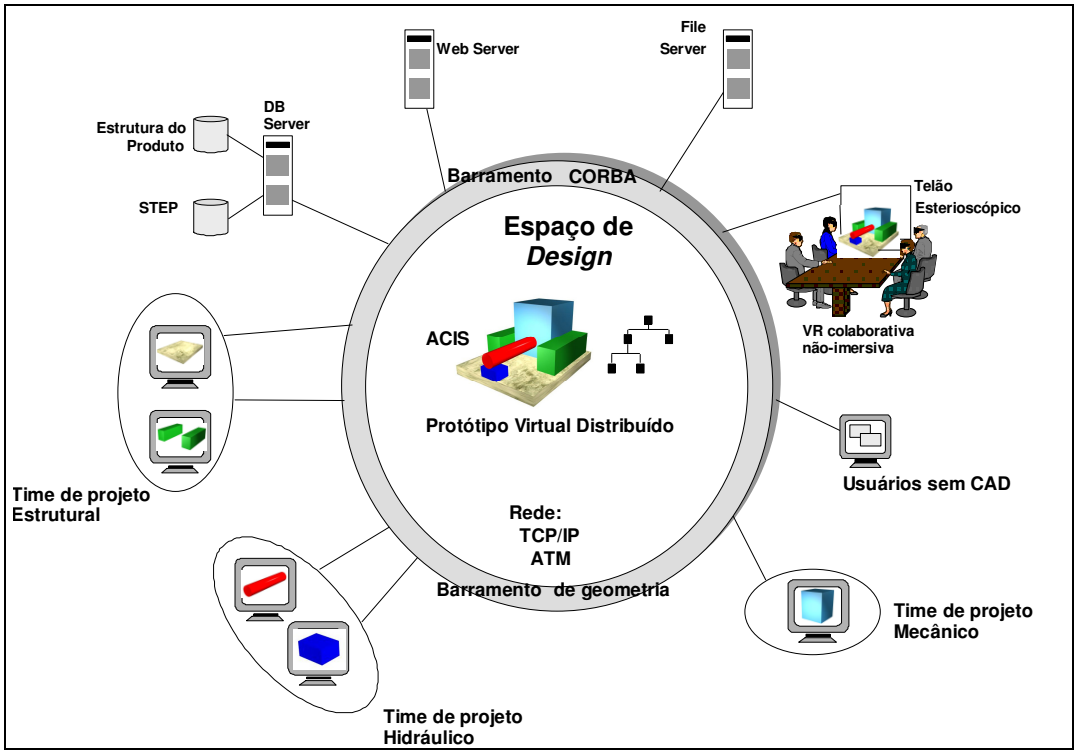


Figura 1: Ambiente para prototipação virtual distribuída.

De uma forma geral, os componentes de um protótipo virtual distribuído devem manter algum tipo de relacionamento entre si e devem reagir a mudanças feitas no ambiente. Neste artigo, a reatividade entre objetos que compõem o protótipo virtual é modelada através do conceito de agentes. A maioria das implementações de sistemas de CAD Inteligente é fortemente baseada em representações simbólicas do mundo. Neste contexto, o paradigma simbólico clássico de Inteligência Artificial apresenta algumas desvantagens tais como ineficiência computacional, incapacidade de lidar com eventos imprevisíveis e inabilidade para lidar com computação interativa. Esta situação é ainda mais complexa no âmbito da Inteligência Artificial Distribuída.

Neste particular, os autores acreditam que o comportamento inteligente dos objetos, os quais serão chamados “agentes”, é melhor representado por “emergência” ao invés de “inteligência”. Emergência significa que o comportamento inteligente entre os agentes “emerge” da interação mútua entre os mesmos, e também da interação dos agentes com o seu ambiente (FEIJÓ, 1998), (HOLLAND, 1999).

Neste artigo, um agente significa um objeto com suas próprias metas (WOOLDRIDGE, 1995). Os agentes distribuídos na rede da figura 1 podem ser representados por grafos que estabelecem comprometimento entre os agentes (ou relações entre atributos dos agentes), formando um sistema de restrições. O sistema pode ser modelado através de um grafo especial, proposto por Gomes (1999).

A figura 2 apresenta um exemplo simples de protótipo virtual distribuído, composto por vários sólidos. O modelo sólido comumente é chamado de maquete eletrônica. Quatro sólidos mantêm algum tipo de relacionamento entre si. Estes sólidos apresentam características de agentes de software e seus relacionamentos são modelados através de relações matemáticas entre alguns de seus atributos de geometria. Por exemplo, o sólido chamado *box2* pode ter sua altura *h* calculada a partir de uma relação matemática entre os atributos *r*, *t* e *l* dos sólidos *cyl1*, *box1* e *cub1*, respectivamente. Conforme mencionado anteriormente, o protótipo está disperso em uma rede. Assim, cada um dos sólidos pode estar em uma máquina diferente comunicando-se através do barramento CORBA, representado por *ORB1* e *ORB2*.

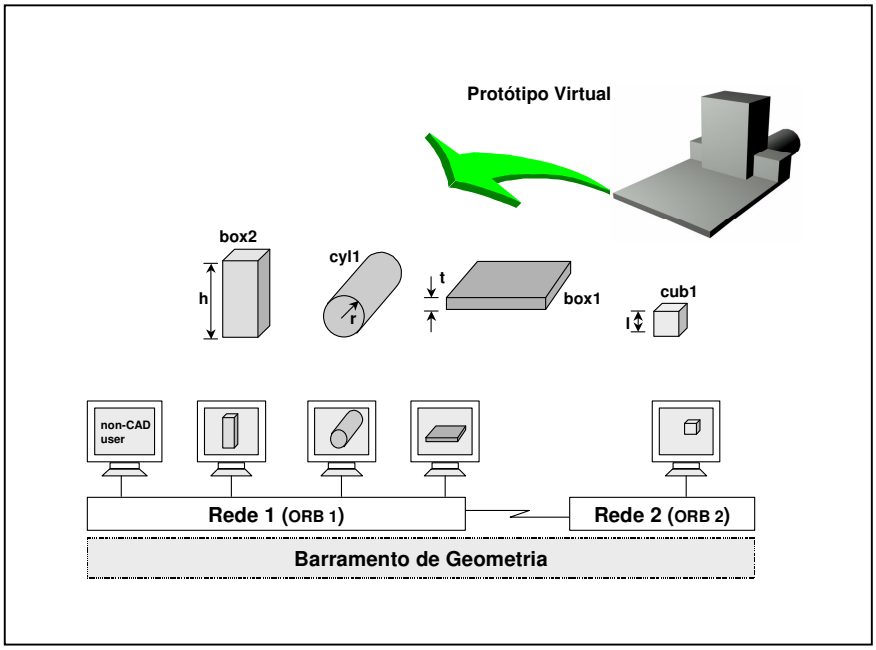


Figura 2: Exemplo de protótipo virtual.

2.1 Grafos Estendidos de Restrições

O Grafo Estendido de Restrições é definido como $G = (V, M, E)$, onde V e M são vértices que representam conjuntos de variáveis e métodos, respectivamente, e E são arestas denotando ligações entre V e M (GOMES, 1998).

Variáveis possuem a forma *obj.attribute* e representam atributos geométricos de sólidos que constituem o modelo 3D da maquete eletrônica. Os métodos representam uma reatividade bidirecional, isto é, uma equação de restrição. Um exemplo de ECG pode ser visto na figura 3.

No processo de design, esse tipo de grafo é construído e modificado em tempo de execução por um usuário ou por um agente qualquer. Por exemplo, a figura 3 ilustra um possível grafo estendido de restrições para a maquete eletrônica da figura 2. *Box1*, *box2*, *cyl1* e *cub1* são agentes representando sólidos com seus respectivos atributos de geometria *t*, *h*, *r* e *l*. As linhas rotuladas com números representam os métodos que implementam os relacionamentos entre os agentes. Pode-se supor que o atual estado da maquete eletrônica é {r = 1, h = 4, l = 2, t = 0.5}. Se um usuário mudar o raio do cilindro *cyl1* para r = 2, o novo estado da maquete virtual será {r = 2, h = 8, l = 4, t = 1}. Alternativamente, um outro usuário poderia ter começado uma nova alteração na maquete, mudando a altura do sólido *box2* durante a execução do algoritmo para cálculo da alteração anterior.

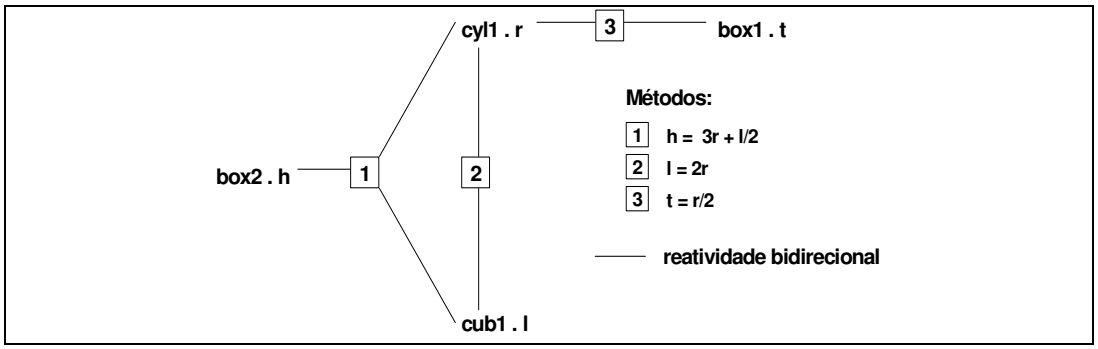


Figura 3: Exemplo de grafo estendido de restrições.

Os métodos para reatividade bidirecional são definidos por três tipos de elementos: (1) uma condição (usualmente uma equação de restrição correspondente aos métodos da figura 3); (2) a inversa dessa condição; e (3) um pedaço de código para cada direção da relação reativa. Por exemplo, a ligação entre *cyl1.r* e *box2.h* é definida pelas seguintes condições e trechos de pseudo-código:

- sentido *cyl1.r - box2.h*:

  se condição r = (1/3(h-l/2)) não for satisfeita, então
  atribua o valor h←3r+l/2 em box2

- sentido *box2.h - cyl1.r*:

  se condição h = (3r+l/2) não for satisfeita, então
  atribua o valor r←1/3(h-l/2) em cyl1

A operação de atribuir um novo valor em um agente deve disparar outras ligações reativas que este agente possa ter. A figura 4 ilustra esse processo bidirecional, de acordo com o grafo da figura 3, para o caso de haver sido feita uma alteração no raio do cilindro *cyl1*. A operação de alteração de um atributo é representada por *add_value*, que só poderá ser efetivada após a satisfação de todas as condições reativas associadas ao atributo, ou seja, quando todos os testes reativos retornarem valor *TRUE*.

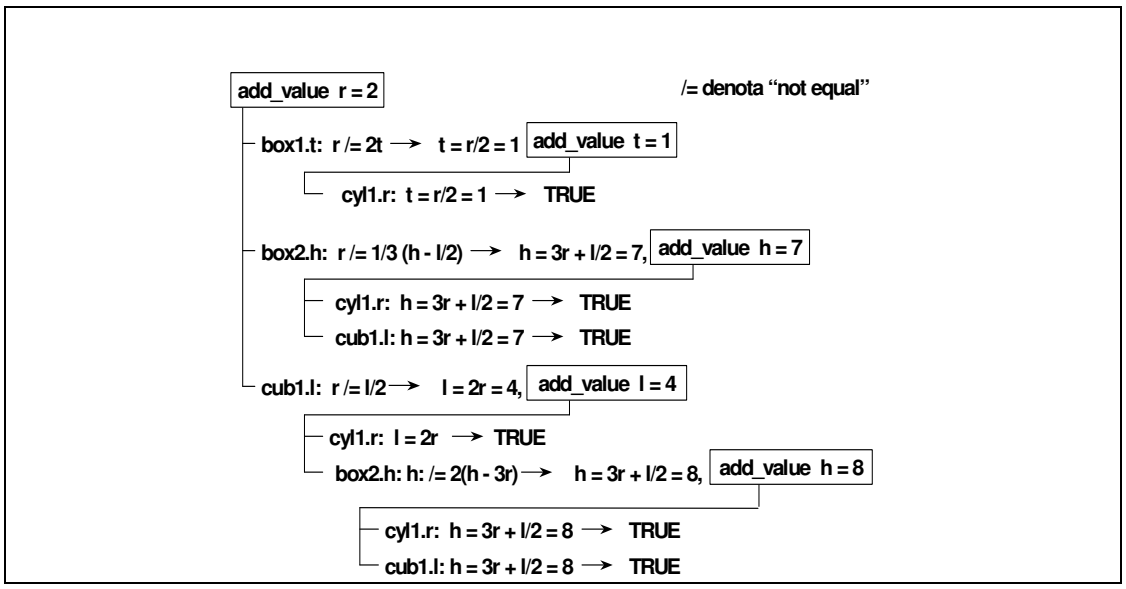

add_value r = 2
/= denota "not equal"
box1.t: r /= 2t → t = r/2 = 1
add_value t = 1
cyl1.r: t = r/2 = 1 → TRUE
box2.h: r /= 1/3 (h - l/2) → h = 3r + l/2 = 7,
add_value h = 7
cyl1.r: h = 3r + l/2 = 7 → TRUE
cub1.l: h = 3r + l/2 = 7 → TRUE
cub1.l: r /= l/2 → l = 2r = 4,
add_value l = 4
cyl1.r: l = 2r → TRUE
box2.h: h: /= 2(h - 3r) → h = 3r + l/2 = 8,
add_value h = 8
cyl1.r: h = 3r + l/2 = 8 → TRUE
cub1.l: h = 3r + l/2 = 8 → TRUE

Figura 4: Um processo reativo bidirecional.

As relações do tipo *link-to* viabilizam a existência da reatividade bidirecional. Cada agente *i* ligado a outros agentes possui uma lista *link-to*$_i$ com membros na forma:

$$< AGT_k.t_n,\ r_{ij}^{kn}>$$

a qual denota uma ligação entre o atributo t_j do agente *i* com o atributo t_n do agente *k*, ou seja:

$$AGT_i.t_j \text{ ---------- } AGT_k.t_n$$

onde r_{ij}^{kn} é definido pela pela função apresentada no quadro 1:

```
função r_ijkn (AGTi, AGTk)
  obtenha atributo Tj do agente AGTi
  obtenha atributo Tn do agente AGTk
  obtenha outros atributos AT1, AT2, … de outros
  agentes, se houver
  se condição(Tj, Tn, AT1, AT2, …) não é satisfeita,
  então
   calcule novo valor de Tn usando a inversa da condição
   retorne add_value(Tn para AGTk)
  fim
  retorne TRUE
fim da função
```

Quadro 1: Função para reatividade bidirecional.

Antes de adicionar um novo valor a um atributo de um agente, a função *add_value* executa cada função *r_ijkn* encontrada na lista *link-to* do agente. Este mecanismo garante a propagação das alterações em todas as direções. É importante salientar que o objetivo no uso dos ECGs não é o mesmo que o encontrado em algoritmos tradicionais de satisfação de restrições. O foco dos ECGs é lidar com reatividade e fazer uma análise de sensibilidade de parâmetros de definição do protótipo virtual, ao invés de promover a resolução de um problema clássico de satisfação de restrições, conforme abordado em Yokoo (2001).

3 Uma proposta de arquitetura para prototipação virtual distribuída

Para implementação de ECGs, os autores propõem uma arquitetura do tipo cliente-servidor, onde a gerência do grafo é centralizada. Neste modelo os agentes de design estão em aplicações cliente que acessam um módulo servidor contendo o ECG. A comunicação entre clientes e servidor é feita através de um barramento CORBA. O CORBA (Common Object Request Broker) é uma arquitetura para interoperabilidade entre aplicações em diferentes máquinas em ambientes heterogêneos (SIEGEL, 1996), (OBJECT MANAGEMENT GROUP, 1995). O CORBA permite que as aplicações se comuniquem independentemente de sua localização ou da linguagem de programação na qual foram implementadas .

A proposta apresentada aqui contempla o uso da linguagem de programação Java (SUN MICROSYSTEMS INC., 2002a), por se tratar de uma linguagem multiplataforma que já nasceu com a visão de redes e objetos distribuídos. Além disso, a arquitetura prevê o uso do Java 3D como ferramenta para desenvolvimento dos sistemas de CAD para modelagem geométrica do protótipo virtual. O Java 3D (SUN MICROSYSTEMS INC., 2002b) faz parte de um conjunto de APIs conhecido como "The Java Media Family" e consiste em um conjunto de mais de 100 classes Java que permite criar e manipular estruturas gráficas em 3D. Na implementação de um protótipo virtual, cada objeto é derivado de uma das classes do Java 3D. A figura 5 apresenta a arquitetura geral proposta.

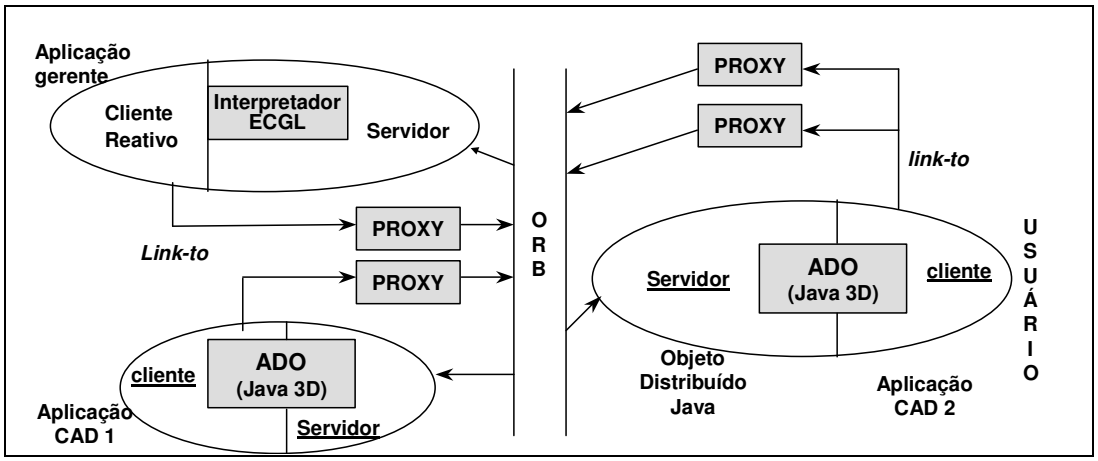


Figura 5: Arquitetura para tratamento de ECGs.

A aplicação gerente é responsável pela manutenção do ECG correspondente ao protótipo virtual distribuído entre as várias aplicações CAD. Estas, por sua vez possuem adaptadores clientes (proxy) para o ORB (*Object Request Broker*), que representa a camada de comunicação CORBA entre as aplicações. Os objetos 3D do protótipo virtual estão encapsulados em classes Java chamadas ADO (*Abstract Data Object*), cuja função é prover uma interface entre as classes Java 3D de geometria e o ORB. Cada objeto distribuído Java é, portanto, um agente de design que também desempenha o papel de objeto servidor CORBA, quando atende requisições de alteração de valores de atributos da aplicação gerente. A aplicação gerente possui um interpretador da linguagem ECGL, utilizada na especificação das variáveis e métodos dos ECGs.

4 A Linguagem de Especificação de Grafos Estendidos de Restrições (ECGL)

Para especificar um ECG usando a Linguagem de Especificação para Grafos Estendidos de Restrições (ECGL) deve-se, inicialmente, relacionar as variáveis que indicam o estado atual da maquete eletrônica e, em seguida, definir as equações de restrição, isto é, os métodos. A especificação formal da linguagem ECGL é apresentada no quadro 2.

```
<ECG>::= VARIÁVEIS <lista de variáveis> MÉTODOS <lista de métodos>
<lista de variáveis>::= <variável> | <variável> <lista de variáveis>
<variável>::= identificador := <valor>
<valor>::= - número | número
<lista de métodos>::= <método> | <método> <lista de métodos>
<método>::= identificador = <expressão aritmética>
<expressão aritmética>::= <termo> <menor prioridade>
<menor prioridade>::= + <termo> <menor prioridade> | - <termo>
                      <menor prioridade> | ε
<termo>::= <unário> <maior prioridade>
<maior prioridade>::= * <unário> <maior prioridade> | / <unário>
                       <maior prioridade> | ε
<unário>::= - <elemento> | <elemento>
<elemento>::= identificador | número | ( <expressão> )
```

Quadro 2: Especificação formal de ECGL.

Sintaticamente, uma variável é definida por um identificador, seguido por :=, seguido por um valor, que deve ser um número real precedido ou não pelo sinal unário (-). E um método é definido por um identificador, seguido pelo símbolo =, seguido por uma expressão aritmética. Observa-se que na especificação apresentada, é levada em consideração a prioridade dos operadores aritméticos. Assim, em ECGL, o grafo da figura 3 é especificado conforme mostrado no quadro 3.

```
VARIÁVEIS
     r := 1
     h := 4
     l := 2
     t := 0.5
MÉTODOS
     h = 3*r + l/2
     l = 2*(h - 3*r)
     l = 2*r
     t = r/2
     r = 2*t
     r = (h - l/2)/3
     r = l/2
```

Quadro 3: Arquivo de especificação de um ECG.

O interpretador ECGL lê da aplicação gerente o arquivo de especificação de um ECG em tempo de execução, e automaticamente estabelece as relações matemáticas entre as variáveis correspondentes aos vértices do ECG. Para tanto, constrói uma tabela onde cada linha contém a variável especificada, o valor atual e as relações correspondentes. Dessa forma, o vértice *cub1.l* da figura 3 tem a seguinte entrada na tabela: (*identificação* = l, *valor* = 2; *relações* = (2*(h-3*r) ; 2*r)).

Após a interpretação do código, a aplicação gerente constrói o grafo correspondente e disponibiliza serviços para disparo de alterações nos valores dos vértices do grafo. As alterações podem ser solicitadas pelas aplicações clientes. A interface da aplicação gerente com o ORB é

feita através de duas funções, *get_value* e *set_value*, que remotamente propagam alterações entre as aplicações do modelo.

Com base na arquitetura proposta, foi implementada uma aplicação exemplo composta por um módulo gerente e vários módulos clientes, os quais contêm apenas atributos numéricos, ao invés de modelos geométricos em Java 3D. Os testes com o protótipo foram realizados em ambiente Windows 2000, utilizando o Borland VisiBroker (BORLAND, 2002) como implementação do barramento CORBA. E na implementação do interpretador ECGL foi utilizado JavaCC (WEBGAIN, 2002).

5 Considerações finais

A prototipação virtual torna-se muito mais interessante quando objetos 3D são agentes dinâmicos distribuídos em rede. Este artigo utilizou os Grafos Estendidos de Restrições (ECGs) como estruturas que viabilizam a interação entre estes agentes. É importante salientar que o objetivo no uso dos ECGs não é o mesmo que o encontrado em algoritmos tradicionais de satisfação de restrições. O foco dos ECGs é lidar com reatividade e fazer uma análise de sensibilidade de parâmetros de definição do protótipo virtual. Foi proposta uma nova arquitetura para sistemas de prototipação virtual distribuída baseada na linguagem Java, na API Java 3D e na definição de uma nova linguagem de especificação para ECGs. A implementação de um sistema completo para prototipação virtual distribuída vai além do escopo deste trabalho. Contudo, a linguagem ECGL representa um avanço nesta direção, pois resolve o problema de especificação dinâmica das relações de restrição entre agentes de design que compõem o protótipo virtual.

Neste trabalho foi dada maior ênfase à implementação do interpretador ECGL e do módulo servidor de resolução de ECGs. Os módulos clientes implementados apenas simulam a existência de aplicações mais complexas de modelagem de sólidos utilizando o Java 3D. A implementação destas aplicações figura entre as próximas etapas deste trabalho de pesquisa.

Existe ainda a necessidade de realização da análise da complexidade do algoritmo para resolução das restrições nos ECGs, visto que o algoritmo apresentado é não-determinístico, isto é, pode levar a solução para valores inesperados. Os autores são de opinião que este é um dos aspectos mais importantes a serem considerados nos trabalhos futuros sobre ECGs. Por fim, sugere-se uma análise mais aprofundada sobre a possibilidade de distribuição do próprio ECG, descentralizando o controle exercido atualmente pelo módulo gerente, conforme foi inicialmente sugerido em Feijó (2001).

6 Agradecimentos

Os autores agradecem ao CNPq pelo apoio financeiro para a realização de parte deste trabalho.

Referências

BORLAND, Borland Corporation. **Borland Enterprise Server, VisiBroker Edition.** [S.l.: s.n.], 2002. Disponível em: <http://www.borland.com/besvisibroker/index.html>. Acesso em: 10 jan. 2003.

FEIJÓ, B.; GOMES, P. C. R.; BENTO, J.; SCHEER, S.; CERQUEIRA, R. Distributed agents supporting event-driven design processes, In: ARTIFICIAL INTELLIGENCE IN DESIGN CONFERENCE, 4., 1998, Lisboa. **Proceedings…**Lisboa: Kluwer Academic Publ., 1998. p. 557-577.

FEIJÓ, B.; GOMES, P. C. R.; SCHEER, S.; BENTO, J. Online algorithms supporting emergence in distributed CAD systems. **Advances in Engineering Software**, England, v. 32, Issues 10-11, p. 779-787, 2001.

GOMES, P.C.R. **Prototipação virtual em modelagem de sólidos distribuída**. 1999. 94 f. Tese (Doutorado em Informática) - Laboratório de CAD Inteligente, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro.

GOMES, P.C.R.; FEIJÓ, B.; CERQUEIRA, R.; IERUSALIMSCHY, R. Reactivity and pro-activeness in virtual prototyping, In: INTERNATIONAL SYMPOSIUM ON TOOLS AND METHODS FOR CONCURRENT ENGINEERING, TMCE '98, 2., 1998, Manchester, UK. **Proceedings…**Manchester: Manchester Metropolitan University, 1998. p.242-253.

HOLLAND, J.H. **Emergence:** from chaos to order. New York: Perseus Books, 1999.

OBJECT MANAGEMENT GROUP Inc. **The Common Object Request Broker architecture and specification; Revision 2.0**. Framingham, MA, USA, 1995.

SIEGEL, J. **CORBA fundamentals and programming.**, New York: John Wiley & Sons, 1996.

SPATIAL TECHNOLOGY Inc. **ACIS 3D Geometric Modeler R11.** Boulder, USA, 2003.

SUN, Sun Microsystems. **Java – J2EE.** [S.l.: s.n.], 2002a. Disponível em: <http://java.sun.com>. Acesso em: 26 mai. 2002.

SUN, Sun Microsystems. **Java3D API.** [S.l.: s.n.], 2002b. Disponível em: <http://java.sun.com/products/java-media/3D/index.html>. Acesso em: 26 jan. 2003.

WEBGAIN, Webgain Corporation. **JavaCC – Java Compiler Compiler.** [S.l.: s.n.], 2002. Disponível em: <http://www.webgain.com/products/java_cc/>. Acesso em: 15 jan. 2003.

WOOLDRIDGE, M.J.; JENNINGS, N.R. Intelligent agents: theory and practice. **Knowledge Engineering Review**, v. 10, n. 2, 1995.

YOKOO, M. **Distributed constraint satisfaction**: foundations of cooperation in multi-agent systems. Tokyo: Springer-Verlag, 2001.

Avaliação da Interface do Auto-Atendimento das Agências do Besc: Uma Abordagem Ergonômica

Vera Schuhmacher Niedersberg Schuhmacher (UNISUL)
shummy@unisul.br

Maria Inés Castiñeira (UNISUL)
minesc@brturbo. com

Aidê Luzia A. de Souza (UNISUL)

Roselane M. M. Camargo (UNISUL)

Resumo. Este trabalho visa pesquisar a adequação do serviço auto-atendimento do Sistema Financeiro Besc - SFBESC, sob a ótica da engenharia de usabilidade da interface com o usuário. Como o serviço de auto-atendimento é basicamente feito de interações homem-máquina os fatores técnicos, e principalmente, humanos devem ser levados em conta para que o usuário possa explorar completamente e da forma mais amigável possível, as potencialidades do software. A partir da compreensão das técnicas de usabilidade selecionaram-se para verificação da interface as técnicas de avaliação prospectiva, a Avaliação Heurística e a análise ergonômica do trabalho.

Palavras-chave: Ergonomia de interfaces, avaliação heurística, técnicas prospectivas, interfaces de auto-atendimento

1 Introdução

Grandes avanços tecnológicos ocorreram nos últimos anos na área de informática, o surgimento constante de novas tecnologias incentivou o aumento do uso dos computadores para realização das mais variadas tarefas. As constantes mudanças não surpreendem, o que há algum tempo era inimaginável, hoje faz parte do dia a dia de muitos. Desde a época dos cartões perfurados até a leitura de códigos de barras e robôs que andam e falam substituindo o homem em muitas tarefas, podem-se perceber os avanços tecnológicos que tornam-se mais ousados a cada dia. Quando o usuário se vê frente a um novo dispositivo interativo, tem expectativas básicas bem definidas. Espera encontrar algo que possa aprender rapidamente, que seja de fácil uso e que, além disso, seja útil em relação a seus objetivos (Cybis, 1995).

Tem-se observado um crescente aumento na preocupação com a interface homem-máquina bem como o reconhecimento da importância da participação do usuário em todo o processo de desenvolvimento do sistema, nem sempre foi assim. Apesar da constante evolução da informática, somente na década de 80 aumentou o interesse pela importância do projeto de interface. A nova realidade na qual os computadores não são utilizados somente por usuários experientes, gerou a necessidade de desenvolver um produto de software com uma interface cada vez mais simples, fácil e de rápida interação. Abriu-se, então, um amplo campo de pesquisa com o objetivo de tornar os sistemas mais "amigáveis", por meio de recursos que auxiliem o usuário na realização das tarefas de comunicação com o computador.

Este trabalho visa a obtenção de informações relacionadas a adequação do serviço de auto-atendimento do Sistema Financeiro Besc - SFBESC, sob a ótica da engenharia de usabilidade da interface com o usuário.

Para atingir sua principal meta, a satisfação do cliente, o SFBESC implantou a automação bancária. Com a automação, o cliente passou a ter informações rápidas e precisas. A automação foi

apenas um passo para que serviços, como o auto-atendimento, se tornassem uma realidade em toda a rede de agências do SFBESC. As máquinas, denominadas terminais de cliente, são dotadas de uma tela que exibe as opções de auto-atendimento e teclado para uso do cliente, ligado ao servidor de rede. Cada rede local é suprida de tantos terminais quantos forem necessários, os quais classificam-se em duas categorias: Terminal de Cliente com Vídeo (TCV) e Caixas Automáticos. Através dos TCVs , os clientes podem efetuar consulta de saldos, extratos de conta corrente, fundos de investimento financeiro, aplicações e resgates nos fundos, pagamentos e transferências. Nos Caixas Automáticos o cliente além de ter acesso aos serviços acima, pode efetuar saque em espécie, sem o uso de cheque.

2 Ergonomia

Segundo Ferreira (1986), "ergonomia é o conjunto de estudos que visam a organização metódica do trabalho em função do fim propósito e das relações entre o homem e a máquina".

Do ponto de vista ergonômico, o homem é visto como o mais importante componente num sistema tecnológico, e não apenas uma parte dele. A mudança de enfoque, visando agora o homem como peça fundamental do sistema de produção, gradualmente alterou conceitos, surgindo o cuidado de adequar o trabalho, o equipamento e o meio ao homem. É necessário que o trabalhador se sinta motivado em seu trabalho, e uma das formas de se conseguir satisfação e maior produtividade é através da ergonomia.

A ergonomia busca estudar aspectos que estão diretamente relacionados com o comportamento do homem no trabalho como: as relações entre o homem, a máquina, o ambiente, a informação, a organização e as conseqüências do trabalho.

Acredita-se que quanto maior a adaptação e a auto-satisfação do homem ao trabalho, maior será seu índice de produtividade

"Os objetivos práticos da ergonomia são a segurança, satisfação e o bem-estar dos trabalhadores no seu relacionamento com sistemas produtivos. A eficiência virá como resultado. Em geral, não se aceita colocar a eficiência como sendo o objetivo principal da ergonomia, porque ela, isoladamente, poderia significar sacrifício e sofrimento dos trabalhadores e isso é inaceitável, porque a ergonomia visa, em primeiro lugar, o bem-estar do trabalhador" Ida apud Gonçalves(1997).

2.1 Ergonomia de Interfaces

Na comunicação via computador, ressalta-se a importância da interface do software, como ponto fundamental na interação homem-computador. Interface, segundo Croft apud Vavassori (1995), é o modo pelo qual o usuário mapeia suas tarefas sobre o conjunto de ferramentas disponíveis em um sistema computacional.

Já Hartson apud Vavassori (1995) define interface com o usuário como todo hardware e software que suporta diálogo entre o sistema e o usuário.

Um conceito muito usado quando se fala de interface é "amigabilidade ao usuário". Amigável ao usuário é a interface que tem a habilidade de reagir como o usuário espera. Pode-se, portanto, considerar o conceito de amigável como sendo definido em termos das características de consistência e satisfação (Dehning apud Vavassori 1995).

A interface deve ser natural e intuitiva. O que se deseja é uma interface que minimize o treinamento necessário, permitindo que o usuário inicie seu trabalho mais cedo. No entanto, a interação do ser humano com o computador não é natural, pois o homem não nasce com a capacidade de operar os dispositivos físicos e lógicos usados na interação com os computadores.

Prova disso é a dificuldade encontrada por grande parte da população para operar terminais bancários.

Natural e intuitiva para o usuário é aquela interface que se comporta de maneira semelhante a interfaces já conhecidas por ele. O conceito de natural e intuitiva está então, ligado a características de consistência ou que obedece a regras formuladas por ele durante os primeiros instantes de interação.

Schneiderman (2002), conceitua Ergonomia de Software como sendo "um conjunto de regras, normas e recomendações utilizadas na concepção de ferramentas através do computador colocadas a disposição do usuário".

Uma regra ergonômica constitui um princípio de concepção ou de avaliação cujo objetivo é obter ou garantir que a interface homem-máquina seja ergonômica. As regras ergonômicas devem respeitar critérios ergonômicos na concepção da interface. Os critérios por sua vez, constituem uma dimensão que nos leva à uma interface elaborada, eficiente, sofisticada, mais cordial e menos sujeita a erro (Scapin, apud Moço, 1996).

2.2 Técnicas de Avaliação

O papel da avaliação de interfaces é garantir que o sistema reúna realmente todos os requisitos necessários a uma interação confortável entre o usuário e a interface. As avaliações oferecem a oportunidade de observar se a interface de um aplicativo está bem construída e integrada ao ambiente de trabalho do usuário.

"A avaliação possui três objetivos principais, a avaliação da funcionalidade do sistema, do impacto da interface no usuário e a identificação de problemas específicos com o sistema. É importante que a funcionalidade esteja de acordo com os requisitos do usuário. O impacto da interface sobre o usuário é avaliado em termos da capacidade funcional do sistema, considerando aspectos como sua real utilidade, a facilidade de aprender a usá-lo efetivamente. As identificações dos problemas específicos evitam que o usuário tenha resultados frustrados ou se sinta confuso" (Silveira, 2001).

Com base nos resultados das avaliações de usabilidade, podemos distinguir três tipos de técnicas de avaliação ergonômica: (a) as prospectivas, que buscam a opinião do usuário sobre a interação com o sistema, (b) as preditivas/analíticas, que buscam prever os erros de projeto de interfaces sem a participação direta de usuários, (c) as objetivas/empíricas, que buscam constatar os problemas a partir da observação do usuário interagindo com o sistema (Cybis ,1997).

2.2.1 Questionários de Satisfação

Esta técnica considerada prospectiva, é baseada na aplicação de questionários e entrevistas, seu objetivo é a verificação da satisfação do usuário em relação ao sistema e sua operação. Essa técnica é muito eficiente, pois o usuário é quem conhece melhor o software e suas interações, podendo apontar defeitos e qualidades em relação ao que o sistema se propõe a fazer (Cybis, 1997).

Os questionários de satisfação realizados por meio de pesquisa de opinião, estão entre as técnicas mais utilizadas, devido ao seu baixo custo, a possibilidade de detectar o grau de falhas e, com isso estabelecer prioridades de projeto, estimando o potencial mercadológico do produto e servindo ainda como complemento para testes futuros de usabilidade (Medeiros, 1999).

2.2.2 Avaliação Heurística

As técnicas analíticas não necessitam da participação direta de usuários nas avaliações. Essas avaliações são baseadas num estudo, feito pelos projetistas ou por um profissional em usabilidade, de versões intermediárias ou acabadas de software interativo (Cybis, 1997).

Conforme Lenat apud Vavassori (1995, p.17), "heurística é uma forma de conhecimento capaz de sugerir ações plausíveis a seguir ou ações implausíveis a evitar". Ou seja , heurísticas são regras informais de julgamento que guiam pessoas numa rápida tomada de decisões. Assim, as heurísticas levam ao objetivo em menor tempo e através de melhores escolhas.

Quando se percebeu que interfaces ruins eram um problema, surgiram propostas de heurísticas para melhorá-las, que variavam de listas curtas de recomendações a listas com mais de mil recomendações. Nielsen e Molich apud Silveira (1997), formalizaram a avaliação heurística. Por meio dela, o avaliador percorre toda a interface, em busca de problemas e obstáculos que os usuários podem encontrar durante a interação, tendo em mãos apenas uma lista de heurísticas (grade de avaliação) de usabilidade como guia (Sears Apud Silveira, 2001).

2.2.3 Ensaios de Interação

As técnicas empíricas dependem da participação direta de usuários, e se referem basicamente aos ensaios de interação e as sessões com sistemas espiões. A preparação de um ensaio requer um reconhecimento detalhado do usuário alvo e das tarefas que ele realiza. Participam dos ensaios pessoas que representam os usuários reais do sistema, que tentam executar as ações típicas de suas atividades, ou seja, é uma simulação do sistema (Cybis, 1997).

A observação do usuário na interação segundo Silveira(2001), é representada por um conjunto de técnicas empíricas que se diferencia por empregar controles experimentais. Ela tem as seguintes características, o usuário participa da avaliação como amostra do público alvo, são usados cenários com tarefas típicas ou críticas e os dados são originados da observação dos usuários durante a interação.

2.3 Análise Ergonômica do Trabalho (AET)

Segundo (Cybis, 1997), a AET lida com uma estrutura de representação do trabalho, tendo claramente uma motivação antropológica. A cultura dos operadores deve ser sempre levada em consideração, de modo a considerar os detalhes (observação constante) e um envolvimento com as estruturas de representação do trabalho procurando entender e conhecer suas dimensões e lógicas. Tarefa e atividade são dimensões que servem como alicerces ou fundamentos da construção do projeto. A tarefa estaria mergulhada em metas e objetivos precisando assim, de organização do trabalho, colheita de dados significativos (entrevistas), informações etc. A atividade é o complemento da etapa da tarefa, onde a pessoa faz uso efetivo das tarefas. Na análise da atividade é fundamental a sucessiva observação do trabalho que está sendo realizado. O ergonomista busca nessas etapas fazer uma análise das informações procurando identificar as "possíveis falhas", as dificuldades e as condições das mais diversas situações.

Na lógica de funcionamento e de utilização, tanto a tarefa como a atividade são os fundamentos reais dessa construção. A lógica de funcionamento tem seu embasamento no conhecimento das funções e de seus mecanismos internos. A lógica da utilização baseia-se nas repercussões visíveis do sistema, (Cybis, 1997). O autor prevê, segundo a metodologia clássica da AET, três etapas de análise a análise da demanda, a análise da tarefa e a análise da atividade.

A análise da demanda deve ser feita pelos analistas por meio de discussões e a partir daí apresentar futuras pretensões acerca da AET. São focos de discussões em reuniões: identificação, objetivos, métodos e limites, onde está situado o trabalho e as fontes de informações. O planejamento da análise é fundamental, necessitando de uma fundamentação consistente, métodos de trabalho bem definidos e um cronograma para as atividades.

A análise da tarefa está centrada na preparação, operação e manutenção de um sistema, sendo realizada através das etapas já mencionadas anteriormente sobre tarefa. São fundamentais nessa análise: a visão dos gerentes, o posto de trabalho, o reconhecimento (conhecimento) do usuário, o

reconhecimento (conhecimento) da tarefa. São elementos importantes na visão dos gerentes: a representação do sistema, os dados organizacionais e os dados econômicos.

Na análise do posto do trabalho é preciso levar em consideração: o ambiente, as ferramentas de apoio e o fluxo de informações. Para conhecer o usuário, devem ser considerados dados como idade, sexo, faixa salarial e o nível de conhecimento (formação, qualificação, experiência, etc). Para conhecer a tarefa é preciso um detalhamento dos seguintes elementos: objetivos, relações, métodos, etapas e condições.

Na análise da atividade deve ser prevista a realização de observações "in-loco" do trabalho do usuário. O diagnóstico é importante, pois é a partir dele que se estabelecem critérios para a tomada de decisão e quando necessárias revisões embasadas nos registros feitos.

3 Resultados da Pesquisa

Com o objetivo de definir o perfil dos usuários dos caixas eletrônicos do Besc e verificar o nível de satisfação do mesmo em relação a interface do sistema de auto- atendimento, aplicou-se a técnica de avaliação prospectiva através do método Questionário de Satisfação.

3.1 Questionários de Satisfação

No estudo foram envolvidos dois avaliadores, o tempo estimado de aplicação foi de dez horas por avaliador envolvido. Estabeleceu-se no estudo a aplicação de questionários para dois usuários alvo genéricos:

- Usuário "Gerente" (Responsáveis pelo auto-atendimento nas agências e funcionários da área de informática);
- Usuário "Caixa eletrônico" (usuários que fazem uso de caixas eletrônicos nas agências do Besc).

Para o usuário Gerente foram entrevistados sete funcionários responsáveis direta ou indiretamente pelo auto-atendimento, cujas áreas de trabalho são as de informática, atendimento ao publico (auto-atendimento) e administração. Constatou-se que existem funcionários que trabalham no auto-atendimento que não possuem conhecimento em informática. Quanto ao grau de instrução observou-se que todos possuem o ensino médio ou ensino superior. Destacam-se a seguir algumas considerações feitas pelo usuário gerente:

- A interface do caixa apresenta rótulos de opções e opções de menu sem significado claro, deixando o cliente frustrado e inseguro durante a utilização. Exemplo: opção PAGAMENTO → IPVA/DAR;
- opções consideradas importantes são apresentadas em pontos não tão visíveis ao usuário. Exemplo: a opção SAQUE apresenta a possibilidade de saque de um valor definido pelo cliente, a opção OUTRO VALOR. O gerente aponta a dificuldade do usuário em ler essa opção;
- opções destrutivas apresentadas sem confirmação e com telas idênticas as opções mais usadas. Exemplo: CANCELAMENTO DE CARTÃO;

Para o usuário Caixa eletrônico, foram entrevistados oitenta e oito clientes, usuários do auto-atendimento, em datas e locais diferentes. A diferenciação entre datas e locais deu-se com o intuito de atingir um público bem diversificado. Uma vez que entre esses dias é feito pagamento dos funcionários públicos ativos e inativos, além do movimento diário normal. Em relação ao grau de instrução, 83 % dos entrevistados possuem ensino médio e superior. Quanto ao conhecimento em

informática 66% dos entrevistados afirmaram ter domínio sobre o assunto. Independente do grau de instrução a maioria utiliza os terminais de 2 a 10 vezes no mês.

Ao estabelecer um comparativo observou-se que o Saque é a opção mais usada para 63 dos usuários, o Extrato é a segunda opção mais usada para 44 usuários e a opção Saldo é a primeira opção mais usada para 8 usuários.

Observou-se que nenhum dos entrevistados utiliza a opção Outros Serviços. Os usuários com 1º grau fazem pouco uso de opções como Transferências e Pagamentos e não utilizam a opção Fundos. O público alvo encontra-se em uma faixa etária onde 1,14% representam menores de 15 anos, 9,1% entre 15 e 20 anos, 67% entre 20 e 50 anos e 23% são usuários acima de 50 anos. Apesar do público alvo ser formado na sua maioria por pessoas com uma faixa etária média, bom nível de instrução e com algum conhecimento em informática, pode-se verificar que 56% dos entrevistados já tiveram algum problema ao usar o auto-atendimento. Vale observar que essa pergunta só foi inserida após a primeira reformulação do questionário. Sendo assim, dos oitenta e oito entrevistados, somente sessenta foram indagados a este respeito. Resultando em vinte oito entrevistados com a resposta omitida . Os problemas apontados encontram-se no gráfico apresentado na figura 1.

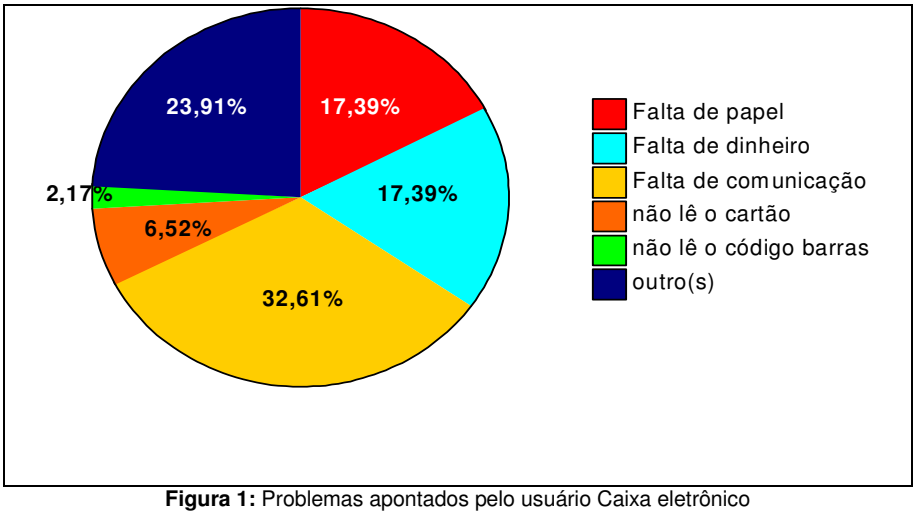


Figura 1: Problemas apontados pelo usuário Caixa eletrônico

No gráfico da figura 1 é possível observar que os problemas comentados pelos usuários da amostra refletem-se na condução do sistema. Nas situações apresentadas o usuário não é avisado pelo sistema de forma preventiva de ocorrências de exceções ou erro. A opção de saque, por exemplo, permite que o usuário complete toda a operação e apenas no final, depois de estabelecida a confirmação o sistema informa ao usuário a insuficiência de dinheiro no caixa, este problema e responsável por 17,39 % da amostra de problemas observados na pesquisa.

3.2 Resultados da Avaliação Heurística

A avaliação heurística realizada neste trabalho fundamentou-se nos critérios ergonômicos de Bastien e Scapin (1993). A avaliação foi aplicadas às telas de Saque, Extrato, Saldo e Pagamento do Sistema de Auto Atendimento. Foram avaliadas setenta e uma (71) telas, que compõem a seqüência de ações necessárias para a realização dessas tarefas.

Observou-se na análise que dos oitenta e cinco (85) problemas encontrados, cinqüenta e um (51) se repetiam na maioria das telas. Trinta e quatro (34) apareceram de forma isolada, ou seja,

ocorrem em apenas uma tela. O quadro abaixo apresenta uma amostra de alguns dos problemas gerais encontrados na interface, subdivididos por critério ergonômico:

Presteza
As telas não apresentam títulos.
A mensagem de convite " Escolha a opção desejada" , é a mesma para qualquer opção (Saldo, Extrato, Saque, Pagamento), confundindo o usuário que deve lembrar qual a opção inicial digitada.
O convite para o usuário iniciar a transação não é significativo para o usuário. Ex:"Passe o cartão no local indicado" . Qual local?, como passar? Ex.: "Coloque a conta a ser paga na área de leitura e aguarde..." não diz onde é a área de leitura.
3.1.1.1.1.1 Agrupamento por formato
Não existe consistência na escolha das cores para diferenciar as classes, por exemplo : classe mensagens de erro, informação e convite à entrada de dados.
As opções exclusivas não estão separadas. Ex1. opção "Outros Bancos" "Outros Carnes" Ex2. opção "Para escolher outro valor". Essas opções deveriam estar diferenciadas logicamente.
3.1.1.1.1.2 Legibilidade
Tamanho das fontes pequeno (botões) para usuários com problemas visuais.
A fonte dos botões de opções é apresentada em caixa alta, a caixa alta e baixa facilita a leitura
3.1.1.1.1.3 Concisão
As denominações não são breves, ex. EXTRATO INTEGRADO – C/C E POC
3.1.1.1.1.4 Ações mínimas
Rótulo está muito distante do campo exigindo carga mental do usuário para relacionar o campo ao rótulo.
No preenchimento de campos de dados, o usuário após o preenchimento do campo deve teclar <ENTRA> , o que poderia ser automático no preenchimento.
Em telas de formulário, o usuário deve informar o valor total, soma que a interface poderia apresentar pois os valores individuais são informados.
3.1.1.1.1.5 Densidade Informacional
Convite "Escolha a opção desejada" é prejudicado pelo logotipo do Besc (lado esquerdo) que atrai a atenção do usuário para o logo e não para o convite da tela.
A mensagem de informação na tela que sugere o uso do cartão, é Prejudicado pelo uso do fundo da caixa de texto em amarelo, onde são digitados os dados da conta. O usuário muitas vezes nem percebe a mensagem de uso de cartão .
Apresentação da hora em todas as telas da interface é uma informação irrelevante para o usuário.
Repetição de palavras como "EXTRATO" , "BESC" e "SALDO" nos botões, aumentando a densidade informacional da tela.
3.1.1.1.1.6 Controle do usuário
O ritmo de entrada de dados não é controlado pelo usuário, e sim por Time-out(15 segundos). Isto frustra o usuário pois caso ele não consiga realizar a leitura ou o preenchimento dentro deste

tempo o sistema cancela a operação e retorna para a tela Inicial
No preenchimento dos dados, o cursor não pula automaticamente para o próximo campo, necessitando pressionar <ENTRA> a cada final de campo.
Não é permitido ao usuário retornar a uma tela anterior. Ao escolher a opção "PARA ESCOLHER OUTRO VALOR", por exemplo, existe apenas a possibilidade de ANULAR e voltar ao menu geral, sendo impossível retornar a tela seletora anterior.
3.1.1.1.1.7 Proteção contra erros
O sistema não apresenta uma mensagem ao usuário informando a falta de papel ou dinheiro.
O sistema não apresenta uma mensagem ao usuário informando que o extrato é cobrado em determinadas situações.
O sistema não possibilita a correção de parte dos dados digitados, ao clicar ANULA, volta ao menu geral
3.1.1.1.1.8 Mensagem de erro
As mensagens de erro não são significativas para o usuário. Ex. "Aguarde, estabelecendo comunicação". Esta é uma mensagem freqüente quando não se está na agência detentora de sua conta.
Na ocorrência de erros o usuário não recebe informações de como solucionar o problema.
As mensagens não são consistentes durante a interface, mudam de uma tela para outra. Ex: Na tela de Saldo Conta Corrente é apresentada a mensagem "Senha inválida" na própria tela, na caixa de texto onde está escrito "Passe o cartão..." Já a tela de Saldo de Poupança , a mensagem "Senha Inválida" é apresentada em uma caixa de diálogo.
3.1.1.1.1.9 Consistência
Os rótulos, não são consistentes . Ex. A opção Ficha de Cobrança e Outros Carnês tratam do mesmo serviço, ou seja, pagamento de documentos compensáveis, só que um é do Besc, e outro é de outros bancos. Deveriam ter o mesmo nome, só diferenciado pelo banco.
A tela de digitação da senha não é padrão com as outras telas de senha do sistema. A localização dos títulos e das informações, ora o título ora a mensagem de convite aparecem na parte superior da tela.
O fundo azul das caixas de texto, é utilizado tanto para títulos quanto para informações.
3.1.1.1.1.10 Consistência
O botão "ANULAR" da máquina possui funções diferentes no sistema, ora ele apenas apaga os dados de entrada de um campo, ora ele volta a tela inicial.
Inconsistência nas telas que solicitam o uso do cartão. Na tela de saque possui uma caixa com um texto de mensagem entre o título e o convite "Passe o cartão no local indicado", já as telas de pagamentos não apresentam mensagem.
3.1.1.1.1.11 Significado dos códigos
Titulo "confirmação de cartão" não é significativo para o usuário, já que trata-se da confirmação dos dados, por motivo de segurança, para efetuar a operação .
As abreviaturas não são significativas Ex.: IPVA, D.A.R-SC

Quadro 1. Problemas Gerais

O gráfico da figura 2 nos mostra que a interface do sistema atual apresenta como maior problema a quebra de consistência entre telas para situações idênticas. O critério presteza, expressivo no gráfico aponta em sua maioria situações em que o usuário não é conduzido durante a tarefa no sistema.

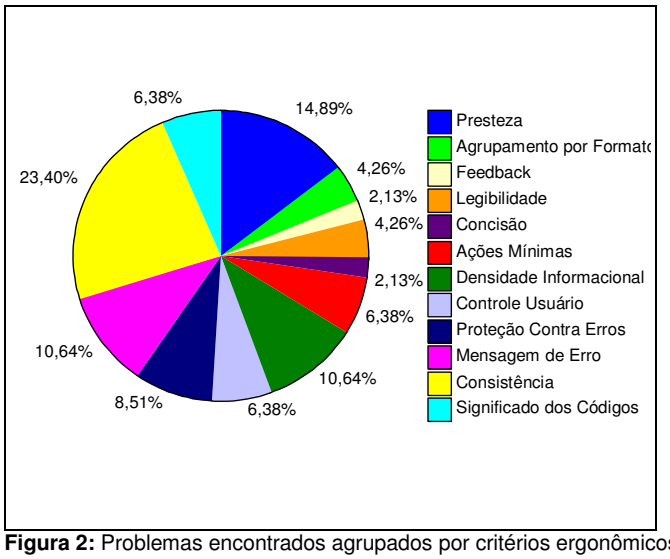


Figura 2: Problemas encontrados agrupados por critérios ergonômicos

O terceiro problema mais apontado durante a avaliação foi a qualidade das mensagens de erro que em muitas situações não facilitam nem auxiliam o usuário na interação e solução do seu problema.

3.3 Análise Ergonômica do Trabalho

Fazem parte da metodologia da AET – Analise Ergonômica do Trabalho, a análise da demanda, a análise da tarefa e a análise da atividade.

3.3.1 Análise da Demanda

Na Análise da Demanda procurou-se a identificação do problema a ser estudado, usuários potenciais e seus objetivos. Nessa fase estudou-se a situação do trabalho e formularam-se as seguintes hipóteses:

- Um pequeno número de usuários, considerando-se o número de clientes do banco, utiliza o auto-atendimento em função da dificuldade de utilização;
- a adequação das tarefas ao usuário, identificando sua maneira de pensar e sua seqüência cognitiva em relação a interação, podem facilitar a interação com o sistema de auto-atendimento;
- a melhoria do conteúdo das telas podem levar a finalização das ações com uma taxa maior de sucesso e satisfação por parte do usuário.

A partir dos resultados dos questionários definiu-se a demanda do projeto. Observou-se que a parcela mais relevante da amostra questionada é formada por usuários de ensino médio e superior e possuem conhecimento em informática. A faixa etária de boa parte da amostra está entre 20 e 50 anos. As opções mais utilizadas no sistema são Saque, Extrato e Saldo.

3.3.2 Análise da Tarefa

Para efetuar esta análise é fundamental conhecer a tarefa, e para isso é preciso um detalhamento dos objetivos, procedimentos, métodos (meios técnicos) e condições temporais das

mesmas. Durante a análise da tarefa referenciaram-se situações em operações de normalidade. Para a definição das tarefas necessárias para alcançar o objetivo utilizou-se o Método Analítico Descritivo (MAD). A seguir será apresentada uma amostra da análise realizada por meio da Tarefa Saque.

Tarefa Saque:

Objetivo : Retirar dinheiro de uma conta corrente, que pode ser de qualquer agência do BESC ou de qualquer Banco pertencente a RVA (Rede Verde Amarela - convênio firmado entre os Bancos Estaduais do Brasil).

Pré Condição :

- Usuário: Ter o cartão, a senha do cartão e possuir saldo suficiente.
- Caixa Eletrônico : estar abastecido com notas suficientes para disponibilizar o saque e estar em comunicação com o sistema.

Procedimentos : A seqüência de ações necessárias para efetuar um saque está descrita abaixo.

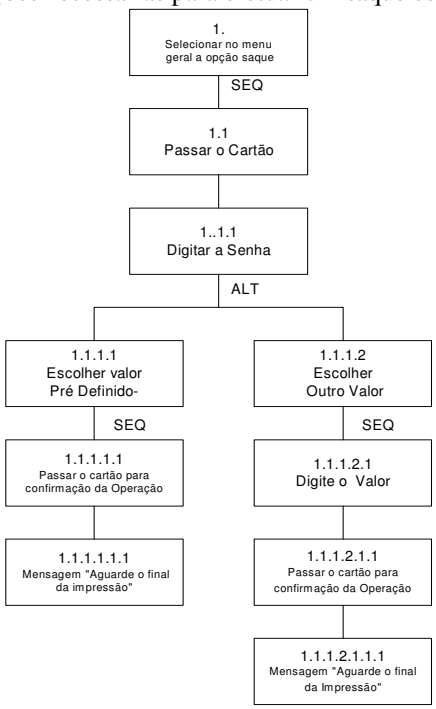


Figura 3: Descrição da Tarefa Saque na Interface Existente

Métodos : leitora de cartão magnético e/ou teclado.

Condições Temporais : O tempo médio da operação, da seleção a impressão do recibo de saque, para um usuário experiente no sistema, é de 30 segundos para valores pré-determinados e 38 segundos quando ocorre digitação de outro valor.

3.3.3 Análise da Atividade

Na análise da atividade deve ser prevista a realização de observações "in-loco" do trabalho do usuário. Pode-se prever registros em situações de normalidade onde são feitas observações contínuas procurando abranger toda a duração do trabalho. O mais importante nesta situação é

conseguir verificar o grau de dificuldade na realização das atividades. A partir do que foi levantado na etapa de análise da tarefa, algumas situações podem ser consideradas problemáticas ou críticas, esta observação se dá confrontando a descrição da tarefa prescrita com a atividade realizada, tentando problematizar. Situações de erros e incidentes apesar de difíceis de serem observadas podem ser feitas por meio de simulações. Após esta simulação apresenta-se ao usuário os procedimentos para a recuperação da situação conflituosa, tentando alcançar a situação de normalidade.

No texto a seguir é apresentada uma situação exemplo identificada como crítica na tarefa saque.

Ação do Sistema:

Falta de Dinheiro - O cash opera com cédulas de dois valores diferentes acondicionados em cassetes separados, quando as cédulas de um desses cassetes termina, automaticamente a mensagem que informa as cédulas disponíveis e os valores pré-definidos de saque são alterados. Se o usuário tentar sacar uma quantia que o terminal não possui, o sistema apresenta a mensagem "Operação Cancelada" e volta ao menu inicial. Isto deixa o usuário frustrado com o sistema deixando-o com a impressão de que o sistema debitou o valor mas não lhe forneceu o dinheiro.

Reação do usuário:

O usuário pensa que o saque não concluído foi debitado na conta emite um extrato ou saldo de sua conta corrente, para confirmar se houve o débito do valor.

4 Conclusões

A Ergonomia pode ser definida como "a ciência da configuração do trabalho adaptado ao homem, e tem como objetivo, o desenvolvimento de bases científicas para a adequação das condições de trabalho às capacidades e realidades da pessoa que trabalha" (Grandejan, 1998).

Com este estudo percebeu-se que o desenvolvimento de interfaces ainda é uma atividade que consome grande quantidade de tempo e esforço da equipe de projeto. Este grande esforço e seus prazos quase sempre pequenos retiram do projeto um dos atores essenciais ao processo: o usuário.

O uso de técnicas de avaliação ergonômicas permitiram a percepção dos problemas enfrentados por este usuário, muitas vezes esquecido, ao fazer uso da interface. O reconhecimento de problemas e sua possível solução podem melhorar esta interação, mesmo que em uma fase tão tardia do modelo de desenvolvimento do software. Por outro lado permitiu a habilitação dos profissionais envolvidos no projeto a utilizarem-se de conceitos desconhecidos sobre a ergonomia de interfaces. Este reconhecimento trouxe a tona as distorções contidas no projeto e as possibilidades de melhoria das mesmas.

5 Referências

CYBIS, W.A., **Ergonomia de Interfaces Homem-Computador**. Apostila para o curso de Pós-graduação em Engenharia de Produção - UFSC, Florianópolis, 1997. p. 26-39,79-95.

FERREIRA, A.B.H., **Novo dicionário da língua portuguesa.** Nova Fronteira, RJ, 2a Ediçao,1986.

GONÇALVES,C.F.F.(1997), **Ergonomia: Uma ferramenta para a melhoria da qualidade dos serviços** In. Anais do 8 Congresso Brasileiro de Ergonomia, Florianópolis, SC.

MEDEIROS, M. A. **ISO 9241: Uma proposta de utilização da Norma para avaliação do grau de satisfação de usuários de software.** 1999. Dissertação (Mestrado em Engenharia de Produção). Programa de Pós Graduação em Engenharia de Produção, Universidade

Federal de Santa Catarina, Florianópolis.

MOÇO, S. S., **O uso de cenários como uma técnica de apoio para avaliações ergonômicas de softwares interativos**.1996. Dissertação (Mestrado em Engenharia de Produção) - Programa de Pós Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, Florianópolis.

SCAPIN, D.L.(1996) **MAF: UNE Métode de Description Des. Tâches**. IN : Colloque Sur e Engeniérie Des Interfaces Homme-Machine, Sophia_Antipolis, France, INRIA

SILVEIRA, M.C. **Checklist Ergonômico. Técnica de Inspeção de Conformidade Ergonômica de Software Interativo voltado à Componentes**. 2001. 84 f. Dissertação (Mestrado em Engenharia de Produção) - Programa de Pós Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, Florianópolis.

SHNEIDERMAN, B. **The Designing the User Interface,** Addison-Wesley, 2002

VAVASSORI, F., **Método Heurístico para Avaliação e Projeto de Interfaces Homem-Computador** . 1995. Monografia (Bacharelado em Ciência da Computação). UCPEL, Pelotas.

Análise comparativa de algoritmos de grafos para um Sistema de Auxílio à Matrícula de Alunos

Rogério Sorroche (FURB)
rs@furb.br

Maurício Capobianco Lopes (FURB)
mclopes@furb.br

Resumo. Este artigo apresenta um sistema para sugestão de horário de matrícula de alunos, implementado com o objetivo de fornecer diversas possibilidades de turmas e horários para o aluno escolher a que melhor se adequa a ele. Para isto foram implementados dois algoritmos de grafos e comparados os seus desempenhos.

Palavras-chave: Grafos, Sistemas Acadêmicos

1 Introdução

Semestralmente a Universidade Regional de Blumenau (FURB) disponibiliza aos seus alunos o procedimento de reserva de vaga. Este procedimento serve como uma pré-matrícula, onde o aluno escolhe as disciplinas que pretende cursar no próximo semestre. A reserva de vaga pode ser feita através da internet, pois já existe um sistema automatizado feito em *Java Applets*. Este *applet* mostra uma lista de todas as disciplinas que são oferecidas ao curso em que o aluno está matriculado, agrupadas por semestre. No sistema, o aluno escolhe a disciplina desejada e automaticamente é mostrada uma grade com os horários semanais da disciplina. Se houver coincidência de horário o mesmo é mostrado com uma cor diferente e uma mensagem ao aluno é emitida, caso contrário, a disciplina pode ser incluída pelo aluno em sua grade.

Para alunos considerados regulares, isto é, que seguem o currículo do curso sem reprovações, o procedimento funciona muito bem. Entretanto, esta não é a regra e muitos alunos têm pelo menos uma ou duas disciplinas nas quais não obtiveram aprovação. Para agravar esta situação, muitas disciplinas são pré-requisitos de outras, ou seja, o aluno tem que obter aprovação nestas disciplinas antes de cursar as outras. Por isso torna-se difícil para o aluno reservar as disciplinas que se encaixem no horário de maneira satisfatória, pois ele tem que passar semestre por semestre, disciplina por disciplina, para verificar se alguma disciplina se encaixa no horário que ele tem disponível. Na grande maioria das vezes ele acaba não fazendo isso.

Neste contexto, surgiu a necessidade de um sistema automatizado para elaborar sugestões de horário na reserva de vaga. Isto envolve selecionar as disciplinas que o aluno está apto a cursar, verificar a disponibilidade de horários do aluno, e, então, elaborar sugestões de horário. Deve-se levar em conta algumas restrições que o próprio aluno pode informar como, por exemplo, dar mais prioridade a uma disciplina do que a outra, ou ainda o fato de que o aluno tem um mínimo de créditos a cumprir todo semestre, e muitas vezes não consegue reservar as disciplinas de modo que este mínimo seja atendido. Assim, o sistema pode beneficiar tanto o aluno, permitindo a ele um melhor aproveitamento de sua grade de horários, quanto à instituição, uma vez que o aluno poderá fazer um maior número de disciplinas possibilitando a conclusão mais rápida do curso.

Problemas de arranjo e otimização como o citado acima, recebem a designação genérica de problemas de *timetabling*, que constituem em construir quadros de opções para uma série de atividades atendendo a um determinado conjunto de restrições. Estes problemas são, em geral, classificados como problemas pertencentes à classe *NP-Difícil*, ou seja, nem sempre possuem algoritmos eficientes para sua execução em tempos de processamento polinomiais. Em outras

palavras, a computação do algoritmo, para um determinado problema, cresce exponencialmente em função do tamanho da instância (Braz, 2000).

Para resolver problemas como estes, surgiram os métodos heurísticos ou aproximativos, definidos, segundo Goldbarg (2000), como algoritmos de busca de soluções em que não existe qualquer garantia de sucesso. Muitos métodos heurísticos já foram aplicados para problemas de *timetabling*, dentre os quais pode-se citar: Algoritmos Genéticos (Braz, 2000) e (Lucas, 2000), *Tabu Search* (Hertz, 1992), *Simulated Anneling* (Thompson, 1996) e algoritmos de Grafos (Schwarz, 1990).

Neste trabalho foram implementados dois algoritmos, baseados na Teoria dos Grafos, para efeito de comparação; o já conhecido Algoritmo Guloso (Szwarcfiter, 1984), e um método baseado no trabalho de Schwarz (1990) que implementou o modelo proposto por Bron e Kerbosh em Bron (1973).

2 Grafos

Segundo Gross (1998), configurações de nós e conexões podem ocorrer em uma grande diversidade de aplicações. Elas podem representar redes físicas, como circuitos elétricos, rodovias ou moléculas orgânicas, e também podem ser usadas para representar interações menos tangíveis, como relacionamentos sociais, bases de dados ou no fluxo de controle em um programa de computador.

Formalmente, estas configurações são modeladas por estruturas combinatoriais chamadas grafos, consistindo de dois conjuntos chamados vértices e arestas e uma relação de incidência sobre elas. Grafos são modelos altamente versáteis para analisar uma ampla gama de problemas práticos em que pontos e conexões entre eles podem ter alguma interpretação física ou conceitual (Gross, 1998). No caso do problema em questão neste trabalho eles são especialmente úteis para representar conjuntos maximais independentes que possam vir a solucionar a alocação das disciplinas baseadas em seus horários.

De acordo com Rabuske (1992), um *conjunto independente maximal* é um conjunto independente para o qual nenhum outro vértice pode ser adicionado sem que destrua a propriedade de independência. Os conjuntos *{a, c, d, f}* e *{b, g}* na Figura 1, são conjuntos independentes maximais. Um grafo qualquer pode conter muitos conjuntos independentes maximais e eles podem ser de diferentes tamanhos. Dentre os conjuntos independentes maximais, normalmente, o que possui maior número de vértices é o de maior interesse.

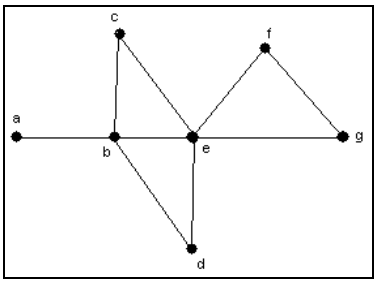


Figura 1 – Conjuntos independentes maximais

Segundo Rabuske (1992), existem diversos métodos para encontrar conjuntos independentes. Muitas vezes, porém o problema é apresentado como sendo o de determinação das cliques maximais de um grafo, pois a cada conjunto independente de um grafo *G* corresponde a uma clique maximal de um grafo complementar *G'*.

A seguir estão descritos dois algoritmos que podem ser utilizados para encontrar conjuntos independentes em um grafo:

a) algoritmo guloso: possui como característica importante a simplicidade e pode se utilizada para uma infinidade de problemas. Segundo Szwarcfiter (1984), a denominação algoritmo guloso provém do fato de que a cada passo procura-se incorporar à solução até então construída, a melhor porção possível, compatível com algum critério especificado;

b) algoritmo de Bron e Kerbosh: baseado no modelo proposto por Bron e Kerbosh em Bron (1973) que consiste em gerar sistematicamente todas as cliques de um grafo completo formando uma árvore. Schwarz (1990) propôs uma variação que foi utilizada no problema da geração de horários, para encontrar conjuntos independentes maximais e que foi utilizada na implementação deste trabalho. O algoritmo em essência é o mesmo, mas é aplicado sobre o grafo complementar em relação ao grafo completo.proposto por Bron (1973), como um método para encontrar todas as cliques de um grafo. Uma das características do métod apresentado é que ele caracteriza-se como um algoritmo do tipo *Branch and Bound*, em que se tenta várias alternativas abandonando-as a partir do ponto em que se pode prever um insucesso.

3 Apresentação do Sistema

O sistema em questão tem por finalidade apoiar o acadêmico em seu processo de matrícula, gerando sugestões de horários. Para a geração destas sugestões de matrícula, o sistema deve levar em conta vários fatores, entre os quais destacam-se:

a) a necessidade de otimizar o número de disciplinas ou créditos financeiros que o aluno irá cursar, em função da restrição quanto ao número mínimo de créditos que os alunos devem pagar semestralmente. Assim, é desejável que o aluno possa se matricular no maior número possível de disciplinas ou pelo menos em um número que satisfaça este mínimo de créditos;

b) a disponibilidade de horários do aluno para cursar as disciplinas, e sua prioridade em relação às disciplinas que deseja ou precisa cursar em uma determinada fase;

c) o conjunto de pré-requisitos, ou seja, disciplinas que o aluno não pode cursar em função de não ter sido aprovado em determinadas disciplinas anteriores, além de desconsiderar na alocação as disciplinas de fases anteriores do curso em que o aluno já foi aprovado;

d) a possibilidade de o aluno matricular-se em turmas oferecidas a outros cursos, uma vez que em muitos casos a mesma disciplina pode ser oferecida a diversos cursos sendo que o aluno poderá cursá-la desde que encaixe em sua grade de horários.

Sendo assim, o gerador de sugestões de matrícula, deve procurar alocar um conjunto de disciplinas em uma tabela de horários, levando em conta as restrições apresentadas acima. Além disso, o aluno poderá optar por otimizar a resposta do sistema para obter o máximo de créditos possíveis ou o máximo de disciplinas possíveis.

É importante ressaltar que as informações consultadas neste sistema já existem e são cadastrados através do sistema de Registros Acadêmicos da Graduação (RGRA), também referido como sistema acadêmico, desenvolvido e mantido pelo Núcleo de Informática da FURB.

3.1 Especificação do algoritmo para sugestão

O problema de geração das sugestões de matrícula pode ser caracterizado como uma busca em grafos. Assim, foi utilizada a busca em árvore, recorrendo-se a técnicas heurísticas para encontrar soluções em tempo razoável. O trabalho foi baseado no modelo proposto por Schwarz (1990).

3.1.1 Definição do Grafo

O grafo *G(V, E)* formado para a solução do problema foi definido como segue:

a) *V* é um conjunto de vértices que representam as opções de disciplinas que podem ser alocadas. Cada vértice $v \in V$ é uma opção de disciplina que pode ser alocada para o aluno;

b) *E* é um conjunto de arestas que representam as restrições. Cada aresta $e = (v, w)$ representa uma restrição de incompatibilidade de alocação mútua das opções *v* e *w*.

O conjunto *E* de restrições pode conter dois tipos de restrições:

a) coincidência de horários: as disciplinas alocadas não podem ter aulas no mesmo horário;

b) coincidência de disciplina: uma vez que uma disciplina pode ser oferecida a várias turmas e o aluno pode se matricular em qualquer uma delas, uma sugestão de matrícula não pode conter mais de uma vez a mesma disciplina, mesmo que seus horários não sejam coincidentes.

3.1.2 Solução Viável e Solução Ótima

Como solução viável para o problema estabeleceu-se que dado o grafo *G(V, E)*, conforme definido previamente, deve-se assegurar que duas opções de alocação incompatíveis não façam parte da solução ao mesmo tempo. Para tanto, a condição de que quaisquer dois vértices do subconjunto *S* de opções de alocação nunca sejam adjacentes, deve ser satisfeita. Em outras palavras, *S* deve ser um conjunto independente.

Para se garantir que o maior número possível de disciplinas seja alocado, *S* também deve ser um conjunto independente maximal, ou seja, não pode haver uma solução que contenha um subconjunto das disciplinas de uma outra solução gerada anteriormente.

Uma das condições a serem satisfeitas na busca de uma solução ótima para o problema é procurar alocar as disciplinas de acordo com a disponibilidade e a preferência do aluno. Outra condição é a de que o aluno pode optar por otimizar o número de disciplinas ou o número de créditos financeiros alocados. Para satisfazer estas condições foram utilizados coeficientes de custo nos vértices ou disciplinas do grafo. O curso c_{ik} reflete o desejo do aluno em cursar a disciplina *i* no horário *k*. Há também o coeficiente de custo c_i que indica que a disciplina i_1, dependendo do seu número de horas/aula ou créditos financeiros, é mais promissora do que a disciplina i_2. A Tabela 1 apresenta a definição do cálculo destes coeficientes de custo.

Item	**Peso**	**Opções**	**Custo**
Prioridade atribuída pelo aluno a cada disciplina (c_{ik})	10000	Obrigatória	1
		Desejável	2
		Normal	3
		Aceitável	4
Disponibilidade de horários do aluno (c_{ik})	100	Disponível	0
		Aceitável	nº horas da disciplina aceitáveis para o aluno / total de horas de todas as disciplinas
Tipo de Otimização (c_i)	1	Número de Créditos Financeiros	nº de créditos máximo – nº de créditos da disciplina
		Número de Disciplinas	nº de horas da disciplina

Tabela 1 – Definição dos custos das arestas do grafo

O custo c_{ik} determina a prioridade que o aluno irá atribuir a cada disciplina, onde, quanto maior a prioridade menor será o custo. Neste custo está somado também um valor correspondente à disponibilidade de horários do aluno. O custo c_i está associado ao tipo de otimização que o aluno deseja fazer na geração das sugestões. Quando o aluno optar por otimizar para obter o maior número de créditos financeiros, as disciplinas que tiverem o maior número de créditos terão o

menor custo. Caso o aluno opte por otimizar para obter o maior número de disciplinas, as disciplinas que tiverem o menor número de horas terão um custo menor.

Estes custos representam uma penalidade associada a cada disciplina, ou seja, quanto maior o custo menos desejável será a alocação desta disciplina.

Cada vértice *v* do grafo tem então um custo associado. Por extensão, a todas as soluções viáveis pode-se associar um custo total. A solução viável de menor custo é a solução ótima do problema.

3.1.3 Processo de Alocação

O processo de busca da solução é um procedimento de geração de subgrafos, visando encontrar conjuntos independentes. Também os conjuntos gerados serão maximais, o que quer dizer que, nenhuma sugestão gerada estará contida em uma solução gerada anteriormente.

A idéia fundamental é explorar a árvore formando um conjunto independente que a cada nível recebe uma nova opção de disciplina, satisfazendo a todas as restrições do problema. O processo continua até alcançar um determinado nível onde:

a) não existem opções restantes;

b) nenhuma das opções restantes apresenta condições de ser agregado ao conjunto em formação, sem vir a repetir conjuntos anteriormente formados.

Para determinar novos conjuntos, retrocede-se a níveis anteriores, a partir dos quais a busca tem prosseguimento. Este processo se repete até que se retroceda ao nível inicial, quando a busca estará terminada.

Para o procedimento de busca, tanto o algoritmo guloso, quanto o algoritmo proposto por Bron e Kerbosh (Bron, 1973) e modificado por Schwarz (1990), foram implementados. No entanto, o algoritmo de Bron e Kerbosh apresentou um tempo de resposta menor (ver resultados e discussão). No anexo 1 é apresentada a implementação do algoritmo guloso e no anexo 2 é apresentada a implementação do algoritmo de Bron e Kerbosh.

3.1.4 Estratégias para aperfeiçoar a busca

De acordo com Schwarz (1990), algumas estratégias podem ser adotadas para melhorar a performance do processo de busca. As que foram adotadas neste trabalho são:

a) ordenação dos vértices: as opções de alocação de disciplinas são ordenadas em ordem crescente de acordo com o seu custo associado;

b) escolha dos vértices: adotou-se o critério de escolha dos vértices de menor custo em cada nível.

3.2 Implementação

Neste tópico são apresentadas algumas telas do sistema, bem como os detalhes da operabilidade do protótipo implementado. Basicamente, o sistema envolve o fato do aluno elencar um conjunto de disciplinas que deseja incluir nas sugestões, informar as opções que deseja incluir na elaboração (tipo de otimização, restrições e disponibilidade de horários) e então gerar as sugestões. Após a autenticação é apresentada uma tela para o aluno com as opções de seleção das disciplinas para gerar sugestões. Nesta tela o aluno pode optar por descartar as disciplinas em que já obteve aprovação, as disciplinas matriculadas no semestre atual ou as disciplinas em que faltam cursar os pré-requisitos. O aluno também pode optar por deixar o sistema procurar em outros cursos uma disciplina que é ofertada ao seu curso mas em outros horários. Após selecionadas as disciplinas é apresentada a tela da Figura 2.

Nesta tela é apresentado para o aluno o conjunto de disciplinas que foram selecionadas, permitindo a ele definir uma prioridade para cada disciplina. As prioridades que o aluno pode escolher são em ordem crescente: “Aceitável”, “Normal”, “Desejável” e “Obrigatória”.

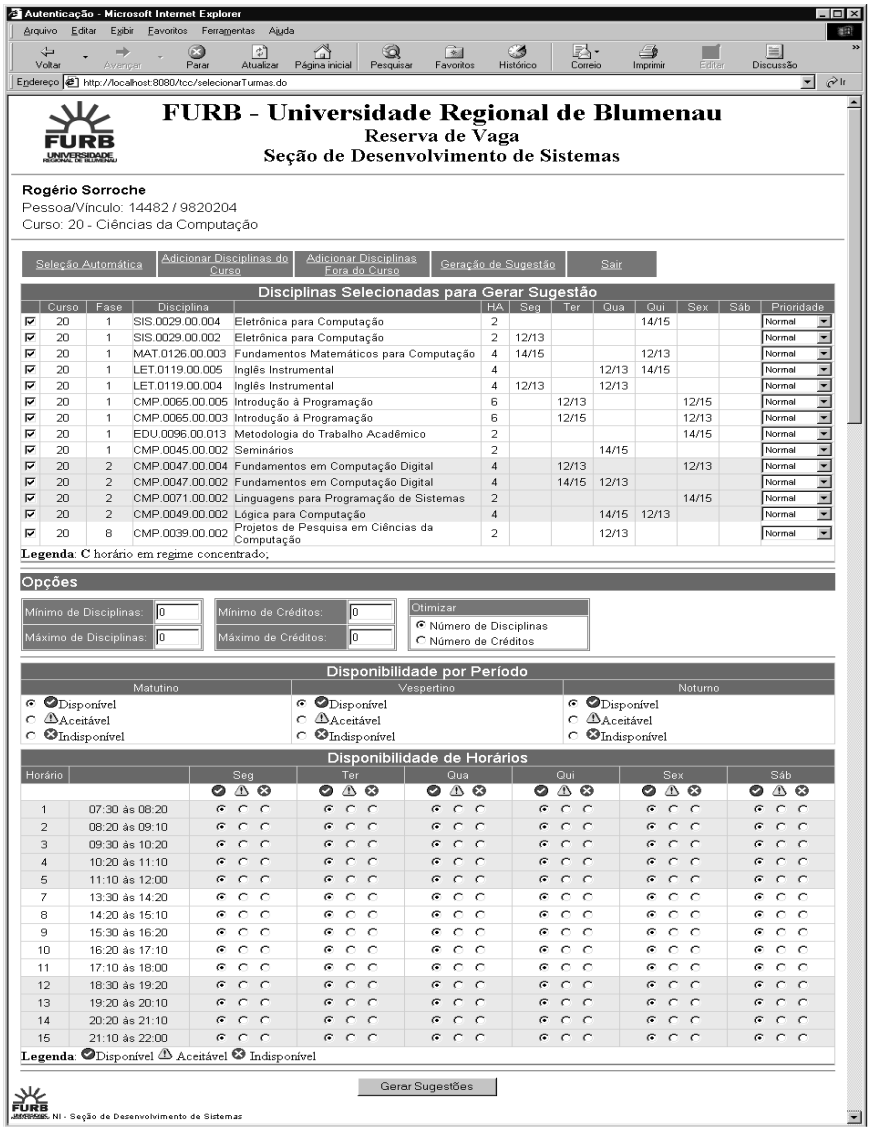

Autenticação - Microsoft Internet Explorer
Arquivo Editar Exibir Favoritos Ferramentas Ajuda
Voltar Avançar Parar Atualizar Página inicial Pesquisar Favoritos Histórico Correio Imprimir Editar Discussão
Endereço http://localhost:8080/tcc/selecionarTurmas.do
FURB - Universidade Regional de Blumenau
Reserva de Vaga
Seção de Desenvolvimento de Sistemas
Rogério Sorroche
Pessoa/Vínculo: 14482 / 9820204
Curso: 20 - Ciências da Computação
Seleção Automática
Adicionar Disciplinas do Curso
Adicionar Disciplinas Fora do Curso
Geração de Sugestão
Sair
Disciplinas Selecionadas para Gerar Sugestão
Curso Fase Disciplina HA Seg Ter Qua Qui Sex Sáb Prioridade
20 1 SIS.0029.00.004 Eletrônica para Computação 2 14/15 Normal
20 1 SIS.0029.00.002 Eletrônica para Computação 2 12/13 Normal
20 1 MAT.0126.00.003 Fundamentos Matemáticos para Computação 4 14/15 12/13 Normal
20 1 LET.0119.00.005 Inglês Instrumental 4 12/13 14/15 Normal
20 1 LET.0119.00.004 Inglês Instrumental 4 12/13 12/13 Normal
20 1 CMP.0065.00.005 Introdução à Programação 6 12/13 12/15 Normal
20 1 CMP.0065.00.003 Introdução à Programação 6 12/15 12/13 Normal
20 1 EDU.0096.00.013 Metodologia do Trabalho Acadêmico 2 14/15 Normal
20 1 CMP.0045.00.002 Seminários 2 14/15 Normal
20 2 CMP.0047.00.004 Fundamentos em Computação Digital 4 12/13 12/13 Normal
20 2 CMP.0047.00.002 Fundamentos em Computação Digital 4 14/15 12/13 Normal
20 2 CMP.0071.00.002 Linguagens para Programação de Sistemas 2 14/15 Normal
20 2 CMP.0049.00.002 Lógica para Computação 4 14/15 12/13 Normal
20 8 CMP.0039.00.002 Projetos de Pesquisa em Ciências da Computação 2 12/13 Normal
Legenda: C horário em regime concentrado;
Opções
Mínimo de Disciplinas: 0
Máximo de Disciplinas: 0
Mínimo de Créditos: 0
Máximo de Créditos: 0
Otimizar
Número de Disciplinas
Número de Créditos
Disponibilidade por Período
Matutino
Vespertino
Noturno
Disponível
Aceitável
Indisponível
Disponibilidade de Horários
Horário
Seg Ter Qua Qui Sex Sáb
1 07:30 às 08:20
2 08:20 às 09:10
3 09:30 às 10:20
4 10:20 às 11:10
5 11:10 às 12:00
7 13:30 às 14:20
8 14:20 às 15:10
9 15:30 às 16:20
10 16:20 às 17:10
11 17:10 às 18:00
12 18:30 às 19:20
13 19:20 às 20:10
14 20:20 às 21:10
15 21:10 às 22:00
Legenda: Disponível Aceitável Indisponível
Gerar Sugestões
NI - Seção de Desenvolvimento de Sistemas

Figura 2 – Tela resumo disciplinas selecionadas e opções

Também, é oferecido ao aluno a opção de restringir os números mínimo e máximo de disciplinas ou créditos, nas sugestões geradas. O aluno pode optar por gerar sugestões que maximizem o número de disciplinas ou o número de créditos. Nos itens “Disponibilidade por Período” e “Disponibilidade de Horários”, o aluno pode informar a sua disponibilidade em determinados horários ou períodos.

Quando o aluno pressionar o botão “Gerar Sugestões”, o sistema irá tentar gerar até dez sugestões e apresentá-las na tela da Figura 3.

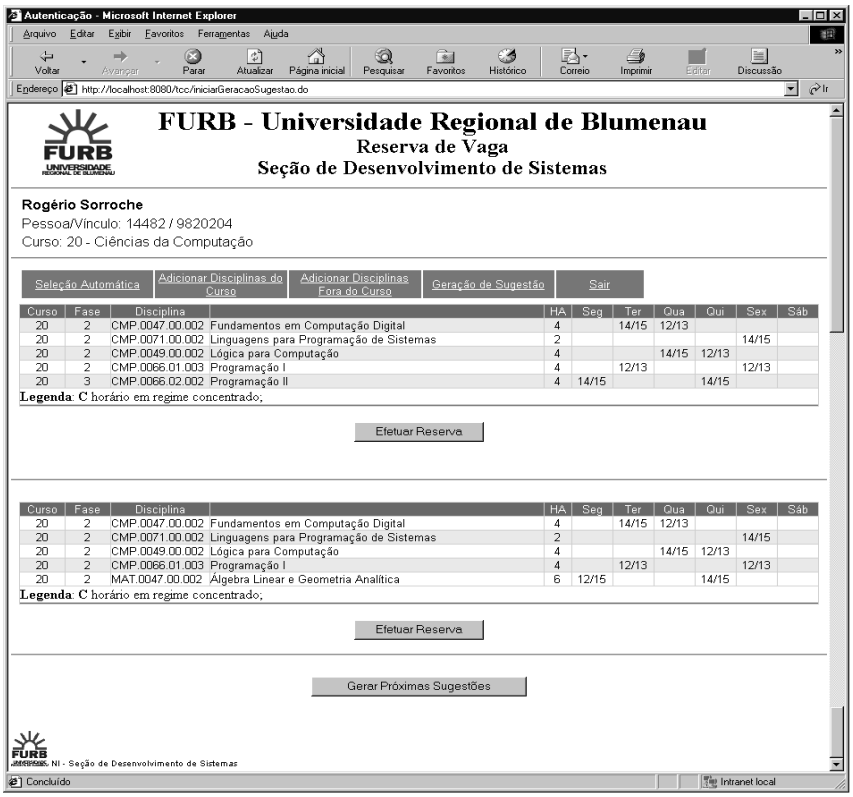


Figura 3 – Sugestões geradas

Esta tela apresenta, então, as sugestões geradas onde o aluno pode efetuar a reserva de acordo com a sugestão ou pode tentar gerar mais sugestões se for possível, através do botão "Gerar Próximas Sugestões".

4 Resultados e Discussão

Para avaliar os algoritmos de geração das sugestões, foram realizados testes utilizando um grafo composto pelo conjunto de disciplinas oferecidas para um determinado curso. Inicialmente, construiu-se o grafo com um total de 52 vértices e foram geradas todas as sugestões possíveis para este grafo. Após o término do processo foram eliminados cinco vértices aleatoriamente do grafo, iniciando-se o processo de geração das sugestões novamente e assim sucessivamente até não haverem mais vértices no grafo.

Os testes foram realizados em um microcomputador PC/AT equipado com um microprocessador *AMD Athlon* de 1,3 Ghz e 512 Mb de memória RAM. O sistema operacional utilizado foi o *Microsoft Windows NT Workstation* versão 4.0. A máquina virtual *Java* utilizada para a execução dos testes foi a versão 1.3 fornecida pela *Sun Microsystems*. A execução dos testes foi realizada localmente. Neste ambiente, somente os serviços necessários ao sistema operacional estavam sendo executados e a máquina utilizada não estava sendo acessada por nenhum cliente.

Para cada teste realizado, foram cronometrados o tempo das primeiras dez sugestões e o tempo total para gerar todas as sugestões. A Tabela 2 apresenta os resultados obtidos.

		Algoritmo Guloso			Algoritmo de Bron e Kerbosh		
		Tempo (ms)			Tempo (ms)		
Vértices	Sugestões Geradas	Primeira Sugestão	Total	Média	Primeira Sugestão	Total	Média
7	5	1	1	2	1	1	0,20
12	11	1	10	0,91	1	1	0,20
17	17	1	60	3,53	1	10	0,59
22	29	10	100	3,49	1	10	0,34
27	49	10	411	8,39	1	40	0,82
32	102	10	1.837	18,01	1	140	1,37
37	166	10	5.388	32,46	1	181	1,09
42	248	10	17.115	61,02	1	330	1,33
47	354	60	40.668	114,88	10	661	1,87
52	794	70	206.977	260,68	60	1.742	2,19

Tabela 2 – Teste comparativo Algoritmo Guloso X Algoritmo Bron e Kerbosh

A Figura 4 apresenta graficamente o teste comparativo entre os dois algoritmos.

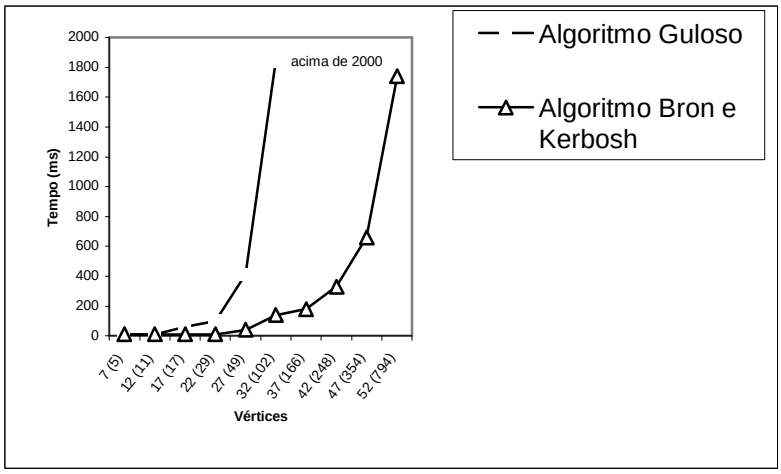


Figura 4 – Gráfico comparativo Algoritmo Guloso X Algoritmo Bron e Kerbosh

Conforme se pode observar, o algoritmo de Bron e Kerbosh apresenta um tempo menor de resposta, especialmente para grafos maiores. No algoritmo Guloso constatou-se que a verificação feita após a geração de uma sugestão para determinar se o conjunto formado é maximal, consome um tempo razoável.

Deve-se ressaltar, no entanto, que ambos os algoritmos se mostram aceitáveis para a geração das sugestões de matrícula, uma vez que geralmente os alunos irão selecionar poucas disciplinas e gerar poucas sugestões. Cabe-se dizer também que os dois algoritmos, obtiveram um tempo médio por sugestão de menos de um segundo, o que considerando se tratar de uma aplicação *web* interativa, é um tempo insignificante.

5 Conclusões

Com o uso do sistema proposto, os alunos que não estejam regulares com a grade curricular do seu curso, têm um valioso e rápido auxílio para se matricularem em um número maior de

disciplinas, atingindo assim o mínimo de créditos financeiros que terão que pagar, dando um melhor aproveitamento à sua grade de horários e reduzindo o seu tempo de curso. Além disso, o sistema beneficia também a universidade, que pode ter um maior retorno financeiro.

Os dois algoritmos utilizados na implementação do sistema mostraram-se adequados, destacando-se o fato de que o algoritmo de Bron e Kerbosh apresentou melhor desempenho nos testes realizados.

Como extensão deste trabalho propõe-se estudar outros métodos de geração de soluções como, por exemplo, *Algoritmos Genéticos*, *Simulated Anneling*, *Tabu Search* ou qualquer dos vários métodos existentes para problemas de *timetabling*, para fazer uma comparação de tempo de execução dos algoritmos e da qualidade das sugestões geradas.

6 Referências Bibliográficas

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML**: **guia do usuário**. Rio de Janeiro: Campus, 2000.

BRAZ, Osmar de Oliveira Jr. **Otimização de horários em instituições de ensino superior através de algoritmos genéticos**. 2000. 144 f. Trabalho de Conclusão de Curso (Mestrado em Engenharia de Produção) – Programa de Pós-Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, Florianópolis.

BRON, Coen; KERBOSCH, Joep. Finding all cliques of an undirected graph. **The Communications of the ACM**, Eindhoven, v. 16, n. 9, p. 575-577, set. 1973.

FURTADO, Antônio Luz. **Teoria dos grafos**: algoritmos. Rio de Janeiro: Editora da Universidade de São Paulo, 1973.

GOLDBARG, Marco Cesar, LUNA, Henrique Pacca L. **Otimização combinatória e programação linear**: modelos e algoritmos. Rio de Janeiro: Campus, 2000.

GROSS, Jonathan e YELLEN, Jay. **Graph theory and its applications**. Nova Iorque: CRC Press, 1998.

HERTZ, A. Finding a feasible course schedule using tabu search. **Discrete Applied Mathematics**. v. 35, p. 255-270, 1992.

LUCAS, Diogo Correa. **Algoritmos genéticos**: um estudo de seus conceitos fundamentais e aplicação no problema de grade horária. 2000. 65 f. Trabalho de Conclusão de Curso (Bacharelado em Informática) – Instituto de Física e Matemática – Universidade Federal de Pelotas, Pelotas.

RABUSKE, Márcia Aguiar. **Introdução à teoria dos grafos**. Florianópolis: Ed. Da UFSC, 1992.

SCHWARZ, Gaston Adair; BARCIA, Ricardo Miranda. **Geração de horário em instituições de ensino com otimização simultânea de tempo e espaço**. 1990. 187 f. Dissertação (Mestrado em Engenharia de Produção) – Universidade Federal de Santa Catarina.

SORROCHE, Rogério. **Sistema de auxílio à matrícula de alunos utilizando *java 2 enterprise edition***. 2002. 108 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

SZWARCFITER, Jayme Luiz. **Grafos e algoritmos computacionais**. Rio de Janeiro: Campus, 1984.

THOMPSON, J.; DOWSLAND, K. A., General cooling schedules for a Simulated Anneling based timetabling system. in: International Conference on the The Practice and Theory of Automated Timetabling, p. 345-363, 1996.

7 Anexo 1 – Algoritmo Guloso

```
package busca;

import java.util.*;

public class AlgoritmoGuloso extends AlgoritmoBuscaSugestao {

    public AlgoritmoGuloso(BuscaSugestao busca) {
        super(busca);
    }
    // procedimento que controla a busca até encontrar uma sugestão
    public NodoBuscaSugestao gerarSugestao() {
        if ( !inicializado ) {
            inicializar();
            inicializado = true;
        }
        NodoBuscaSugestao nodo = null;
        while ( pilha.size() > 0 && nodo == null ) {
            try {
               nodo = fazUmaIteracao();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        return nodo;
    }
    public NodoBuscaSugestao fazUmaIteracao() throws Exception {
        // visita o próximo nodo da pilha
        NodoBuscaSugestao nodo = (NodoBuscaSugestao)pilha.remove(0);
        // se o nodo está nulo gera excecao
        if ( nodo == null ) {
            throw new Exception("Nodo de busca vazio");
        }
        // atribui como nodo  corrente
        setNodoCorrente(nodo);
        // verifica se é o melhor nodo
        if ( nodo.getCusto() > getMelhorNodo().getCusto() ) {
            setMelhorNodo(nodo);
        }
        // verifica se é uma condição de parada
        if ((nodo.getItensParaAlocar().size() == 0) && (nodo.getItensAlocados().size() != 0))
        {
               // se é uma solução e é uma nova solução
            if ( getBusca().isSolucao(nodo) && !getBusca().contemSolucao(nodo) ) {
                return nodo;
            }
        }
        // gera os sucessores deste nodo
        Collection sucessores = expandeNodo(nodo);
        // para cada operador gerado poe os nós na pilha
        // para cada sucessor adiciona na pilha
        for ( Iterator i = sucessores.iterator(); i.hasNext();  ) {
            // obtem nodo sucessor
            NodoBuscaSugestao sucessor = (NodoBuscaSugestao)i.next();
            // adiciona na pilha
            inserePilha(sucessor);
          }
        // se chegou aqui é porque não encontrou solução
        return null;
    }
    // procedimento que pega um nodo e gera todos os seus sucessores
    public Collection expandeNodo(NodoBuscaSugestao nodo) {
        // cria cópia das listas do nodo para não alterá-los
        Collection itensAlocar = (Collection)((ArrayList)nodo.getItensParaAlocar()).clone();
        // lista de sucessores que será retornada
        Collection sucessores = new ArrayList();
        for (Iterator i = itensAlocar.iterator(); i.hasNext(); ) {
            // pega próximo item de sugestao
            ItemBusca itm = (ItemBusca)i.next();
            // remove este item para que não seja mais pesquisado
            i.remove();
            // clona listas de itens atuais
            List itnsAlocarSucessor   = (List)((ArrayList)itensAlocar).clone();
            List itnsAlocadosSucessor = (List)((ArrayList)nodo.getItensAlocados()).clone();
            // elimina da lista de itens para alocar as que possuem coincidências
            eliminarCoincidentes(nodo, itm, itnsAlocarSucessor, true);
            // adiciona item na lista de itens alocados
            itnsAlocadosSucessor.add(itm);
            // cria nodo de busca
            sucessores.add( new NodoBuscaSugestao(getBusca(), itnsAlocarSucessor
                                    , itnsAlocadosSucessor, nodo.getProfundidade()+1) );
        }
        return sucessores;
```

```
    }
    // procedimento de inicialização do algoritmo
    private void inicializar() {
        // cria pilha
        pilha  = new ArrayList();

        // cria nodo inicial
        NodoBuscaSugestao nodo = new NodoBuscaSugestao(getBusca()
                                                       , getBusca().getItensPreprocessados()
                                                       , new ArrayList(), 0);
        // atribui como nodo corrente e melhor
        setNodoCorrente(nodo);
        setMelhorNodo(nodo);
        // coloca nodo inicial na pilha
        pilha.add(nodo);
    }

    // insere nodo na pilha de acordo com a profundidade e custo
    private void inserePilha(NodoBuscaSugestao nodo) {
          boolean inserido = false;
          for (int i = 0; i < pilha.size() && !inserido ; i++) {
              NodoBuscaSugestao n = (NodoBuscaSugestao)pilha.get(i);
              if ( n.getProfundidade() < nodo.getProfundidade() ) {
                      pilha.add(i, nodo);
                      inserido = true;
            } else {
                  if ( nodo.getCusto() < n.getCusto() ) {
                              pilha.add(i, nodo);
                              inserido = true;
                  }
              }
          }
          if ( !inserido ) pilha.add(nodo);
    }
    private boolean inicializado = false;
    private ArrayList pilha;
    private boolean parar = true;
}
```

8 Anexo 2 – Algoritmo de Bron e Kerbosh

```
package busca;

import java.util.*;

public class AlgoritmoBronKerbosh extends AlgoritmoBuscaSugestao {

    public AlgoritmoBronKerbosh(BuscaSugestao busca) {
        super(busca);
    }
    // procedimento que controla a busca até encontrar uma sugestão
    public NodoBuscaSugestao gerarSugestao() {
        if ( !inicializado ) {
            inicializar();
            inicializado = true;
        }
        NodoBuscaSugestao nodo = null;
        while ( pilha.size() > 0 && nodo == null ) {
            try {
               nodo = fazUmaIteracao();
            } catch (Exception e) {
            }
        }
        return nodo;
    }
    public NodoBuscaSugestao fazUmaIteracao() throws Exception {
        // visita o próximo nodo da pilha
        NodoBuscaBronKerbosh nodo = (NodoBuscaBronKerbosh)pilha.get(0);
        // atribui como nodo corrente
        setNodoCorrente(nodo);
        // verifica se é o melhor nodo
        if ( nodo.getCusto() > getMelhorNodo().getCusto() ) {
            setMelhorNodo(nodo);
        }
        boolean backtracking      = false; // indicador de backtracking
        boolean solucaoEncontrada = false; // indicador de solução encontrada
        // CAND vazio e ANT vazio, condicao de parada
        if ((nodo.getItensParaAlocar().size()==0) && (nodo.getItensVisitadosAnt().size()==0))
        {
            if ( getBusca().isSolucao(nodo) ) {
                  solucaoEncontrada = true;
            }
            backtracking = true;
```

```
        // CAND vazio e ANT não vazio, impossível prosseguir
        } else if ( ( nodo.getItensParaAlocar().size() == 0 ) &&
                    ( nodo.getItensVisitadosAnt().size() > 0) ) {
            backtracking = true;
        // verifica se é viável prosseguir (bound), se não faz backtracking
        } else if ( condicaoLimite(nodo.getItensParaAlocar(),nodo.getItensVisitadosAnt() ) ) {
            backtracking = true;
        }
        // se for para fazer backtracking
        if ( backtracking ) {
            // desempilha
            pilha.remove(0);
        } else { // é para continuar estendendo nodo atual
            // estende o nodo atual gerando um novo nodo
            NodoBuscaSugestao novoNodo = estendeNodo(nodo);
            // empilha
            pilha.add(0, novoNodo);
        }
        // solução encontrada, retorna solução
        if ( solucaoEncontrada ) {
            return nodo;
        }
        // chegou aqui porque não encontrou solução
        return null;
    }
    public NodoBuscaBronKerbosh estendeNodo(NodoBuscaBronKerbosh nodo) {
        // pega proximo item CANDIDATO, removendo dos candidatos para não ser mais visitado
        ItemBusca itm = (ItemBusca)nodo.getItensParaAlocar().remove(0);
        // duplica as listas CAND e ANT
        List itnsAlocarSucessor  = (List)((ArrayList)nodo.getItensParaAlocar()).clone();
        List itnsVistAntSucessor = (List)((ArrayList)nodo.getItensVisitadosAnt()).clone();
        // colocar em ANT
        nodo.getItensVisitadosAnt().add(itm);
        // duplica a lista de vértices independentes VI, para facilitar backtracking
        List itnsAlocadosSucessor = (List)((ArrayList)nodo.getItensAlocados()).clone();
        // coloca o item como alocado
        itnsAlocadosSucessor.add(itm);
        // elimina as coincidências
        eliminarCoincidentes(nodo, itm, itnsAlocarSucessor , true);  // CAND
        eliminarCoincidentes(nodo, itm, itnsVistAntSucessor, false); // ANT
        // cria novo nodo de busca
        return new NodoBuscaBronKerbosh(getBusca(), itnsAlocarSucessor
                                        , itnsAlocadosSucessor, itnsVistAntSucessor
                                        , nodo.getProfundidade()+1);
    }
    // procedimento de inicialização
    private void inicializar() {
        // cria pilhas
        pilha = new ArrayList();
        // cria nodo inicial
        NodoBuscaBronKerbosh nodo = new NodoBuscaBronKerbosh(getBusca()
                                                            , getBusca().getItensPreprocessados()
                                                , new ArrayList(), new ArrayList(), 0);
        // atribui como nodo corrente e melhor
        setNodoCorrente(nodo);
        setMelhorNodo(nodo);
        // coloca na pilha
        pilha.add(0, nodo);
    }
    // verifica condição bound ou limite
    private boolean condicaoLimite( List itParaAlocar, List itVistadosAnt ) {
        // se algum vértice de ANT não for adjacente a nenhum em CAND então, limite atingido
        for ( Iterator i = itVistadosAnt.iterator(); i.hasNext(); ) {
           ItemBusca ant = (ItemBusca)i.next();
           boolean adjacente = false;
           for ( Iterator a = itParaAlocar.iterator(); ( a.hasNext() && !adjacente ); ) {
              ItemBusca cand = (ItemBusca)a.next();
              adjacente = ant.isCoincidente(cand);
           }
           // não coincidiu com ninguém retorna true
           if ( !adjacente ) {
              return true;
           }
        }
        // se chegou aqui, não é condição limite
        return false;
    }
    private boolean inicializado = false;
    private ArrayList pilha;
    private boolean parar = true;
}
```

Protótipo de um ambiente de monitoramento e apresentação de programas Java utilizando Reflexão Computacional

Marcel Hugo (FURB)
marcel@furb.br

Romeu Gadotti (FURB)
romeu@wk.com.br

Resumo. Este artigo apresenta o desenvolvimento de um ambiente para monitoração de programas orientados a objeto em Java, baseado em reflexão computacional. Foram estudadas algumas extensões de bibliotecas que possibilitam a reflexão computacional no ambiente Java, tanto em seu aspecto estrutural quanto comportamental, assim como as classes reflexivas da própria Java. Como resultado obteve-se a implementação de um ambiente de representação da estrutura e comportamento das classes utilizadas na aplicação submetida à análise do protótipo.

Palavras-chave: Reflexão computacional; Orientação a Objetos; Java

1 Introdução

Segundo Winblad (1993), existem alguns problemas de entendimento do paradigma de orientação a objetos, o mais comum é a não distinção entre classes e objetos. Classes são gabaritos estáticos que existem somente no corpo de um programa-fonte. Objetos são entidades dinâmicas que aparecem como áreas de memória durante a execução de um programa. Estes problemas de entendimento podem ser parcialmente sanados com a utilização de um ambiente que apresente informações sobre a estrutura das classes de um aplicativo antes do mesmo ser executado, e após sua execução apresente os objetos instanciados e seu comportamento.

Este artigo tem como objetivo apresentar o desenvolvimento de um protótipo de ambiente, que através do monitoramento da execução de um programa orientado a objetos, destaque a estrutura e manipulação dos atributos de uma classe ou instância e a passagem das mensagens entre os objetos deste programa.

O protótipo proposto utiliza-se da tecnologia de reflexão computacional para possibilitar o monitoramento estrutural e comportamental de qualquer programa escrito na linguagem de programação Java. Com a possibilidade do monitoramento do programa Java, torna-se possível o processo automático de representação de seu comportamento durante a execução.

O foco em Java deve-se ao fato de sua freqüente utilização no ensino da tecnologia de orientação a objetos. Por tratar-se de uma linguagem fortemente orientada a objetos e de alta legibilidade, tem-se um aumento significativo na facilidade de entendimento do código-fonte. Entre outras vantagens, como por exemplo, a de ser uma ferramenta gratuita, a linguagem Java destaca-se ainda por oferecer suporte à tecnologia de reflexão computacional.

O artigo introduz alguns conceitos sobre reflexão computacional, sua aplicação em Java, e apresenta o desenvolvimento do protótipo e as conclusões alcançadas.

2 Reflexão computacional em Java

A idéia do paradigma de reflexão computacional não é exatamente nova. Esta idéia originou-se em lógica matemática e recentemente, mecanismos de alto nível tornam o esquema de reflexão um aliado na adição de características operacionais a módulos já existentes (Barth, 2000).

Segundo Senra (2001) o termo **reflexão** remete a dois conceitos distintos no domínio da linguagem natural. O primeiro conceito é reflexão como sinônimo de introspecção, ou seja, o ato de examinar a própria consciência ou espírito. O segundo descreve reflexão como uma forma de redirecionamento da luz. No domínio da Ciência de Computação, reflexão computacional encerra ambas conotações: introspecção e redirecionamento. A primeira denota a capacidade de um sistema computacional examinar sua própria estrutura, estado e representação. Essa trinca de fatores é denominada **meta-informação**, representando toda e qualquer informação contida e manipulável por um sistema computacional que seja referente a si próprio. Por sua vez, a segunda conotação, de redirecionamento, confere a um sistema computacional a capacidade da auto-modificação de comportamento. Ambos conceitos, redirecionamento e introspecção, são tipicamente materializados em linguagens de programação sob a forma de interceptação na execução de primitivas da linguagem. Portanto, a equação resultante é **reflexão computacional = meta-informação + interceptação**.

Segundo o ponto de vista da engenharia de software, reflexão computacional é uma ferramenta de divisão de interesses (*separation of concerns*), sendo assim, esta pode ser usada para permitir que programadores escrevam programas com um grande nível de abstração e com uma boa modularidade (Tatsubori, 2002).

Na programação convencional tem-se uma mistura de programação do aplicativo com complicados algoritmos de policiamento, dificultando assim, a compreensão, a manutenção, depuração e validação do programa. A separação de interesses pode ser vista como a reutilização dos algoritmos básicos de policiamento separados do domínio da aplicação. Desta forma tem-se um meta-espaço, permitindo ao programador focar mais especificamente no domínio da aplicação. Esta funcionalidade adicional é provida através de uma biblioteca de meta-componentes, como será visto mais adiante.

Sempre que o conceito de reflexão computacional for entoado trará consigo expressões como “domínio da aplicação”, “nível base”, “metaníveis” e “arquitetura de metaníveis”, conforme figura 1 (Devegili, 2000). Denomina-se arquitetura de metaníveis qualquer arquitetura de software com características de análise de seu próprio comportamento, sua forma de execução e sua estrutura de dados, além de um nível para o desenvolvimento da aplicação, denominado nível base, e no qual chamadas de procedimento sejam automaticamente desviadas para um outro nível que se preocupe com as análises mencionadas anteriormente. A este nível dá-se o nome de metanível. Uma arquitetura de metaníveis é composta por vários níveis interligados através de um protocolo de comunicação entre os objetos.

Para a realização da técnica de reflexão computacional torna-se necessário o entendimento de duas partes que compõe esta tecnologia, a introspecção (*introspection*), também referenciada como reflexão estrutural (*structural reflection*), que se refere ao processo de obtenção da informação estrutural do programa e sua utilização no próprio programa, e a intercessão (*intercession*), também referenciada como reflexão comportamental (*behavioral reflection*), ou ainda como interceptação, e refere-se ao processo de alterar o comportamento do programa no próprio programa.

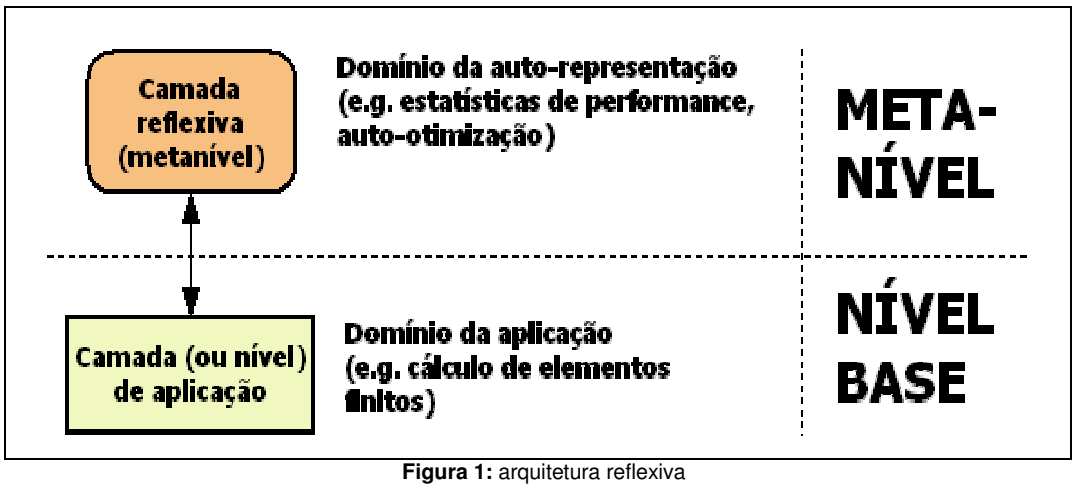

Camada reflexiva (metanível)
Domínio da auto-representação (e.g. estatísticas de performance, auto-otimização)
META-NÍVEL
Camada (ou nível) de aplicação
Domínio da aplicação (e.g. cálculo de elementos finitos)
NÍVEL BASE

Figura 1: arquitetura reflexiva

2.1 Reflexão computacional e Orientação a Objetos (OO)

Apesar do conceito de reflexão computacional ser aplicado sobre diversos paradigmas de programação, Senra (2001) prevê algumas vantagens na união entre reflexão computacional e OO, como a estruturação do meta-domínio e o gerenciamento da complexidade dos metaníveis. O fruto imediato de tal união foi a criação do conceito de metaobjeto.

Um metaobjeto é um objeto, ou seja, possui um estado e um comportamento associados. Além disso, um metaobjeto reside no meta-domínio e está vinculado diretamente a um ou mais objetos que pertençam a uma camada de abstração inferior, como mostra a figura 2 (Devegili, 2000).

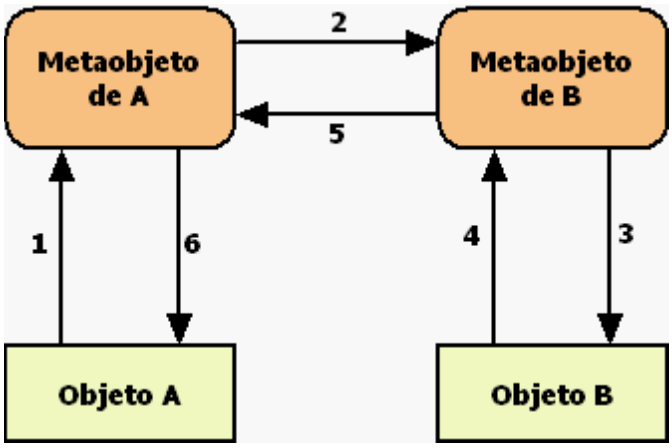


Figura 2: arquitetura reflexiva em orientação a objetos

Barth (2000) diz que reflexão computacional está baseada na noção de definir um interpretador de uma linguagem de programação para a própria linguagem, e que no paradigma de objetos, isto significa representar toda a abstração do modelo de objeto em termos do próprio modelo de objetos. Como conseqüência, a representação de classes, métodos, atributos e objetos são redefinidos por meio de metaclasses e metaobjetos, que estão dispostos em um nível diferente

das classes e objetos. As metaclasses e metaobjetos estão dispostos no metanível, comentado anteriormente.

Contudo, para realizar a comunicação entre o objeto modificado e o objeto modificador, ou seja, entre o nível base e o metanível, é necessário dispor de algum mecanismo. Esta comunicação acontece através de uma interface definida pelo *MetaObject Protocol* (MOP).

Um exemplo prático do funcionamento do MOP está representado na figura 3 (Killijian,1998) e acontece durante a invocação de um método de objeto, feita pelo cliente, e utilizando os parâmetros exigidos (seta 1). O MOP apanha esta chamada, empacota os parâmetros e chama o método do metaobjeto (seta 2). Desta forma o metaobjeto pode executar algumas ações antes de chamar o método do nível-base (seta 4). Após a execução do método de nível base o valor de retorno é empacotado e devolvido ao metaobjeto (seta 2), que novamente, pode executar algumas ações antes de devolvê-lo ao objeto do nível-base (seta 4) que, por sua vez, devolve o resultado da operação ao cliente (seta 3).

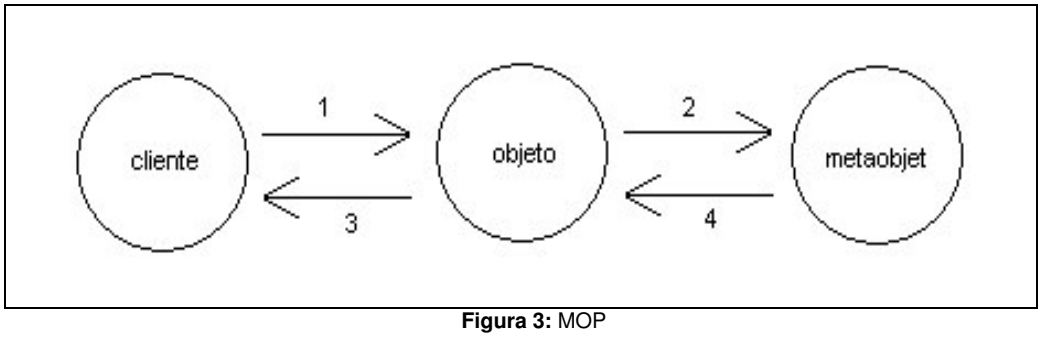


Figura 3: MOP

2.2 *Java.lang.reflect* e Javassist

Java é uma linguagem de programação que suporta reflexão. A habilidade reflexiva da Java dá-se através da API de reflexão, porém é bastante restrita à introspecção (reflexão estrutural). Chiba (1998) afirma que a habilidade da API da Java para alterar o comportamento do programa ainda é muito limitada, ela só permite a um programa a instanciação de uma classe, ou a designação e captura de um valor de campo, ou a invocação de um método.

Na *Java Virtual Machine* (JVM) padrão, todas as classes tem como base a classe *Object*, pré-definida pela linguagem. Segundo Barth (2000), a Java dá suporte para metainformação, permitindo introspecção de classes e objetos através da *package java.lang.reflect*. Entretanto, a máquina virtual Java padrão não dá nenhum apoio direto para protocolos de metaobjetos. Porém, várias extensões da API de reflexão ou da própria JVM foram propostas: OpenJava (Tatsubori, 1998), metaXa ou MetaJava (Golm & Kleinoder, 1998), Guaraná (Senra, 2001) e Javassist (Chiba, 1998).

Javassist, utilizada neste trabalho, foi baseada em uma nova arquitetura de reflexão que pode ser implementada sem modificar o sistema de *runtime* existente, ou seja, sem modificar a JVM. A Javassist é uma extensão da API Java e não é um sistema reflexivo em tempo de compilação, ou seja, não há a necessidade de alterações no código-fonte da aplicação do nível base para que ocorra a reflexão.

Em Javassist, uma classe é representada por um objeto de *CtClass*. O programador que deseja alterar a definição desta classe deve utilizar-se dos métodos do objeto de *CtClass*. Este objeto representa o *bytecode* de uma classe carregada pela JVM. Neste momento obtém-se o acesso à classe, possibilitando que o programa tenha acesso a sua estrutura.

```
CtClass c = new CtClass("Pessoa");
```

Esta linha de código cria um objeto de *CtClass* representando o *bytecode* da classe Pessoa. A classe *CtClass* dispõe de inúmeros métodos para realizar a introspecção e alteração da estrutura da classe. Mudanças na estrutura da classe Pessoa, por exemplo, são refletidas no *bytecode* representado pelo objeto de *CtClass* que carregou a classe.

A informação sobre atributos e métodos é provida através de outros objetos, já que *CtClass* serve apenas para capturar os atributos e métodos da classe. Estes objetos são providos pelas classes *CtField* e *CtMethod* respectivamente. Os objetos de *CtField* são obtidos através do método de *CtClass getDeclaredFields()* e os objetos de *CtMethod* são obtidos através do método de *CtClass getDeclaredMethods()*. Existe ainda a classe *CtConstructor* que mantém as informações sobre os construtores da classe.

A diferença entre a API Javassist e a API de reflexão de Java padrão é que a Javassist provê vários métodos para modificar a estrutura da classe, como mostra a tabela 1. Estes métodos são categorizados em métodos para alterar os modificadores de classe, métodos para alterar a hierarquia de classe e métodos para adicionar novos membros à classe.

Métodos em *CtClass*	Descrição
`void bePublic()`	marca a classe como pública
`void beAbstract()`	marca a classe como abstrata
`void notFinal()`	remove o modificador *final*
`void setName(String name)`	muda o nome da classe
`void setSuperclass(CtClass c)`	muda o nome da super classe
`void setInterfaces(CtClass[] i)`	muda as interfaces
`void addConstructor(...)`	adiciona um novo construtor
`void addDefaultConstructor()`	adiciona um construtor padrão
`void addAbstractMethod(...)`	adiciona um novo método abstrato
`void addMethod(...)`	adiciona um novo método
`void addField(...)`	adiciona um novo campo
Métodos em *CtField*	**Descrição**
`void bePublic()`	marcar o campo como público
Métodos em *CtMethod*	**Descrição**
`void bePublic()`	marca o método como público
`void instrument(...)`	modifica o corpo do método
`VOID SETBODY(...)`	substitui o corpo do método

Tabela 1: métodos para alteração

A API Javassist possibilita a reflexão comportamental (*behavioral reflection*) através de "ganchos" inseridos no programa quando as classes são carregadas. O metaobjeto associado a cada classe mantém métodos de interceptação que podem ser reescritos em classes estendidas da classe *Metaobject* (figura 4).

```
public class MyMetaobject extends Metaobject
{       public Object trapMethodcall(int identifier, Object[] args) {
                // código executado antes do código do nível base}
        public Object trapFieldRead(String fieldName) {
                // código executado antes do código do nível base}
        public void trapFieldWrite(String fieldName, Object value) {
                // código executado antes do código do nível base}
}
```

Figura 4: métodos de interceptação

Atualmente a API da Javassist encontra-se na versão 2.2. Esta versão tornou-se disponível no dia 01 de outubro de 2002 e pode ser adquirida juntamente com sua documentação no endereço http://www.csg.is.titech.ac.jp/~chiba/javassist/survey.

3 Desenvolvimento do protótipo

O protótipo construído neste trabalho monitora a execução de um programa orientado a objetos em Java. Os requisitos identificados para o protótipo foram:

- deverá inicialmente apresentar informações sobre a estrutura das classes de um programa Java;
- deverá também apresentar o comportamento dos objetos durante a execução deste programa, ou seja, apresentar a troca de mensagens dos mesmos.

Para a especificação do protótipo foi utilizada a linguagem UML, através do diagrama de casos de uso, diagrama de classes e diagrama de seqüência. A ferramenta utilizada para a especificação foi o *Rational Rose C++ Demo 4.0.3.*

Na modelagem deste protótipo foram observados dois casos de uso, mostrados na figura 5:

- obter informações das classes: o usuário obtém todas as informações referentes às classes da aplicação que pretende monitorar;
- monitorar a troca de mensagens: o usuário tem a possibilidade de monitorar o comportamento dos objetos da aplicação.

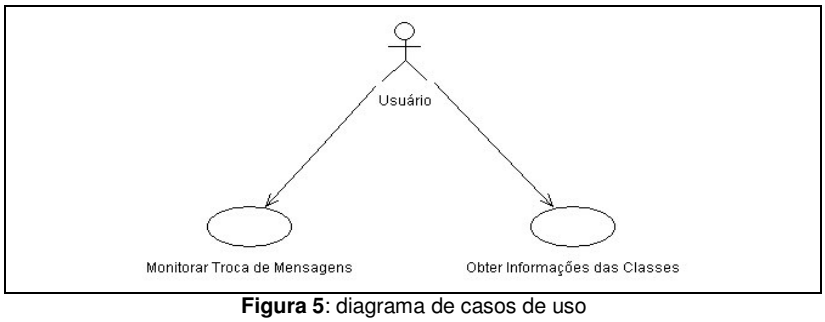


Figura 5: diagrama de casos de uso

Na figura 6 tem-se o diagrama de classes que fornece uma visão geral das classes do modelo proposto, com a seguinte descrição completa:

- Regente: esta pode ser considerada a principal classe do processo de reflexão do programa Java, pois ela é responsável pelo gerenciamento de todas as operações reflexivas como tornar as classes do programa Java reflexivas (prontas para serem

monitoradas) e apresentar todas as informações requisitadas das estruturas das classes do programa submetido ao protótipo;

- Informacao: classe que armazena e seleciona informações referentes às classes informadas pela classe Reflexão. Esta classe manipula diretamente a classe *CtClass* da Javassist obtendo diversas informações como métodos, atributos, construtores, interfaces e outras. Tendo estas informações a classe Informacao as arranja de uma forma mais amigável para manipulação;
- Monitor: responsável pela interceptação dos métodos e alterações em valores dos atributos de objetos das classes do nível base. A classe Regente torna uma classe do nível base reflexiva associando-a a classe Monitor. Assim, sempre que um objeto da classe base for instanciado, um objeto da classe Monitor será instanciado pela Javassist, tornando-se um metaobjeto. Monitor estende a classe Metaobject nativa da Javassist;
- MonitorFrame: esta classe tem a responsabilidade de imprimir todas as mensagens enviadas pela classe Monitor.

As classes *CtClass*, *CtMethod*, *CtField* e *CtConstructor* fazem parte da biblioteca Javassist. Cada uma delas representa uma classe ou um membro dela, como *CtField*, por exemplo, que representa um atributo de uma classe.

A classe *Metaobject* também mencionada no diagrama de classes representa o metaobjeto associado a um objeto do nível base. Este metaobjeto será invocado toda vez que ocorrer uma mudança no comportamento no objeto do nível base associado a ele.

Para cada caso de uso definido foi gerado um diagrama de seqüência correspondente. O primeiro diagrama de seqüência, por ser bastante extenso foi dividido em 3 (três) partes que serão analisadas nas figuras a seguir (Figuras 7, 8 e 9) juntamente com o segundo diagrama.

O monitoramento da aplicação analisada pelo protótipo só torna-se possível após a execução desta primeira etapa de obtenção das informações das classes. Como mostra a figura 7, inicialmente um objeto da classe Regente é instanciado tendo como parâmetro uma *String* que representa o caminho até o diretório onde se encontra o aplicativo. O construtor da classe Regente ao receber esta *String* chama o método setaPath(*String*) que a irá inserir no *ClassPath* da Java. Em seguida o método capturaInformacoes() é chamado na classe Regente. Este chama o método capturaArquivos(), da própria classe, que possui a função de capturar todos os arquivos *.class* do diretório passado no construtor e também instancia um objeto da classe Informacao. Na seqüência o método classesCapturadas() é invocado pela interface na classe Regente. Este irá buscar o nome das classes capturadas no diretório e as retornará em um vetor de *Strings*.

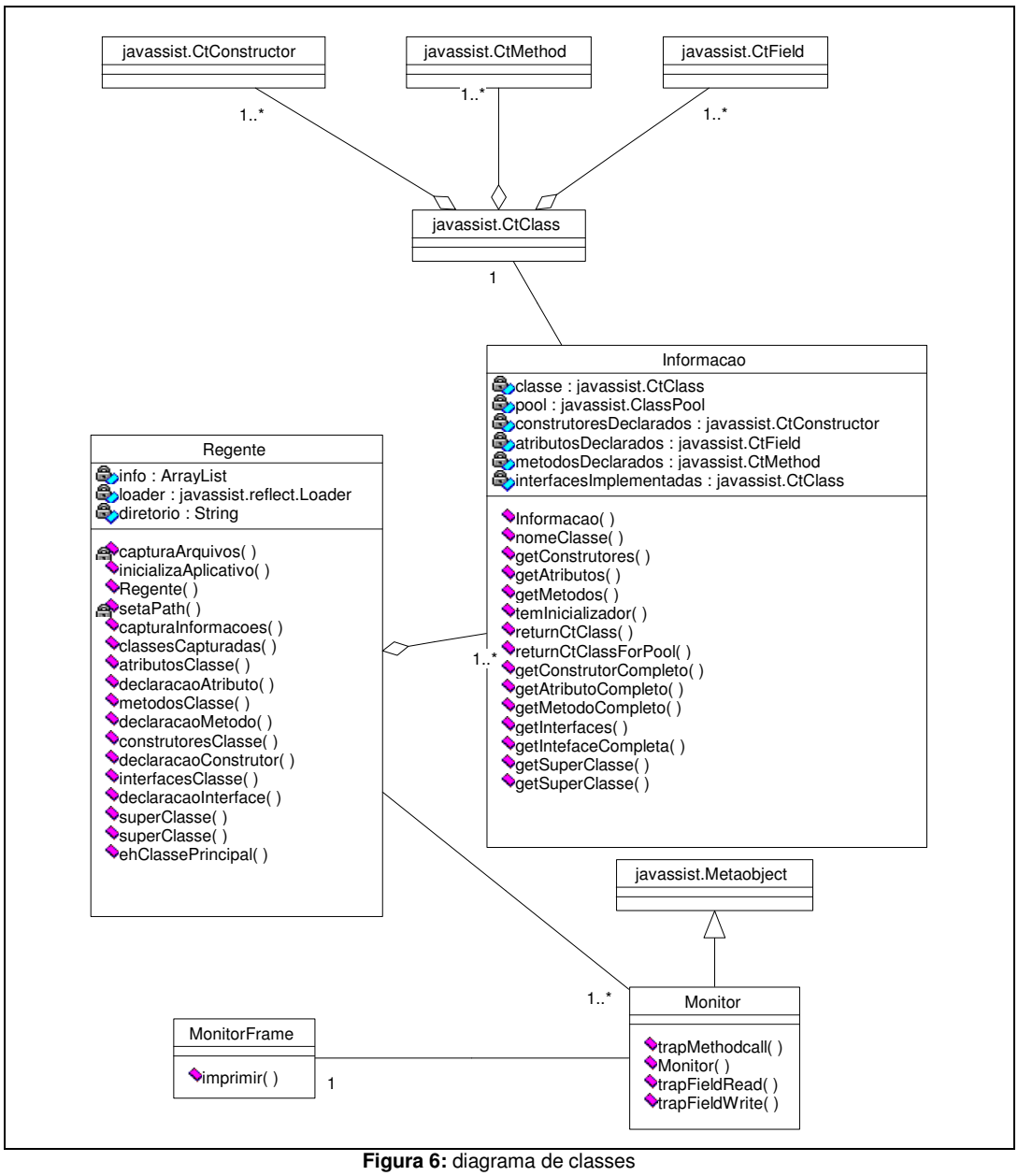


Figura 6: diagrama de classes

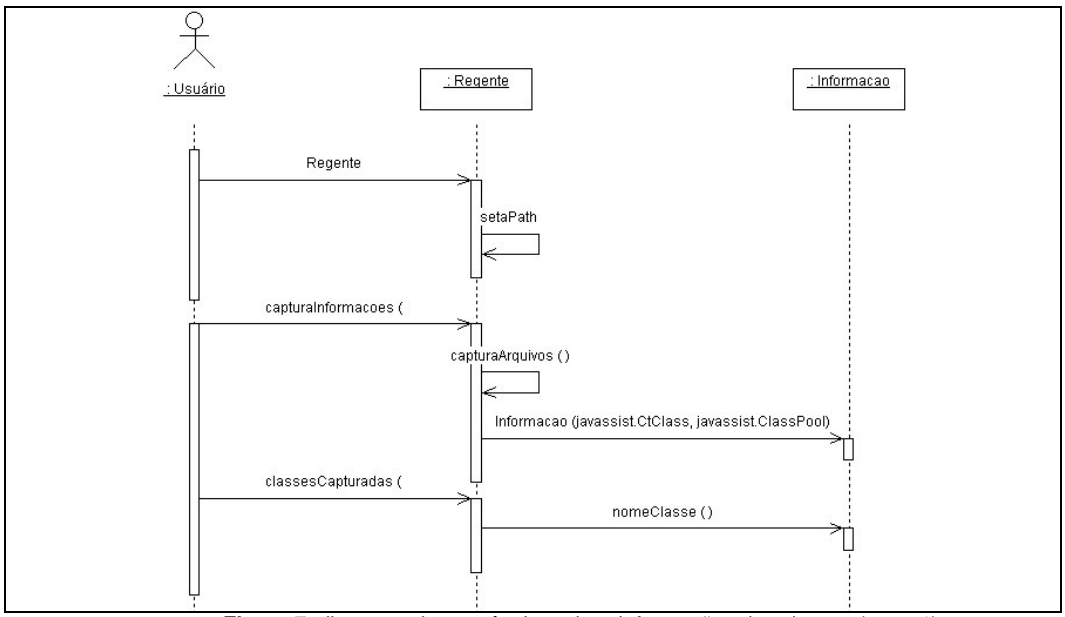


Figura 7: diagrama de sequência – obter informações das classes (parte 1)

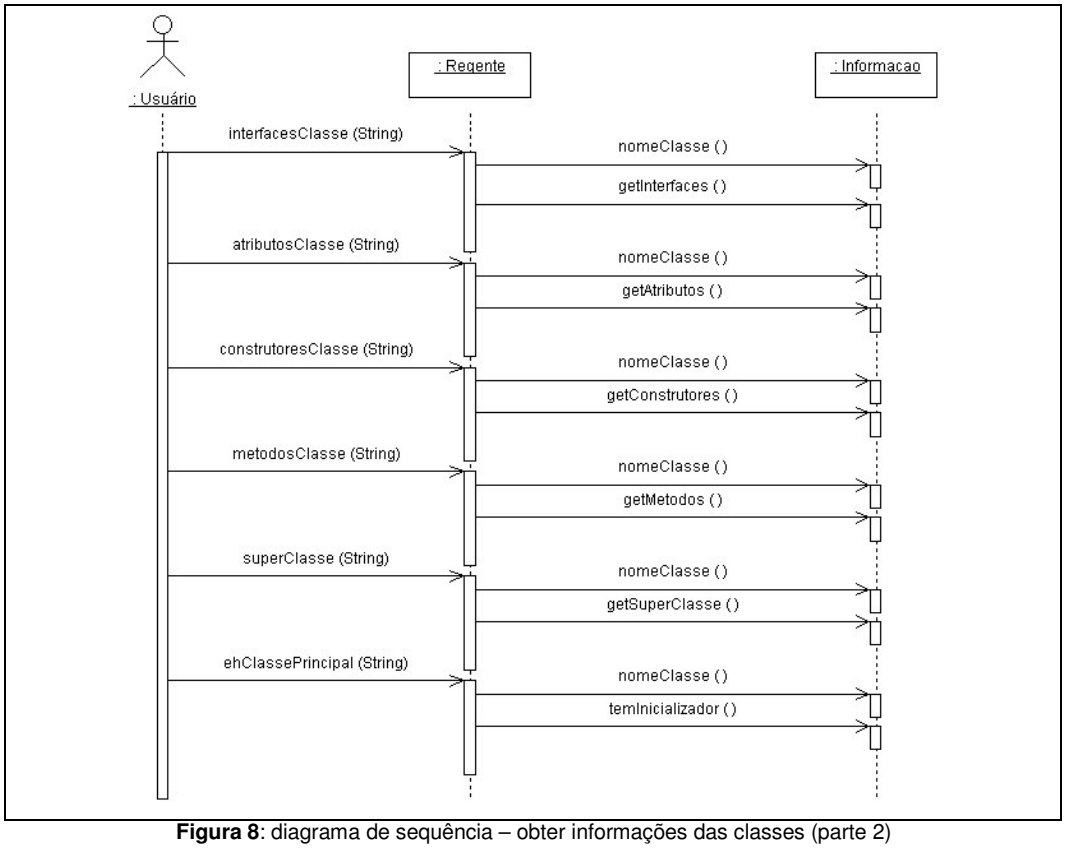


Figura 8: diagrama de sequência – obter informações das classes (parte 2)

Nesta parte do diagrama (Fig. 8) a interface requisita as interfaces, atributos, construtores, métodos e superclasse da classe selecionada pelo usuário. Os métodos funcionam de forma idêntica: primeiro o método da classe Regente é invocado, interfacesClasse(*String*) por exemplo, trazendo consigo o nome da classe como parâmetro. Este irá buscar o objeto da classe Informacao correspondente à classe passada como parâmetro e as interfaces implementadas por ela. A única exceção é o método ehClassePrincipal(*String*) que tem a finalidade de verificar se a classe cujo nome foi passado como parâmetro tem o inicializador de classe *main*.

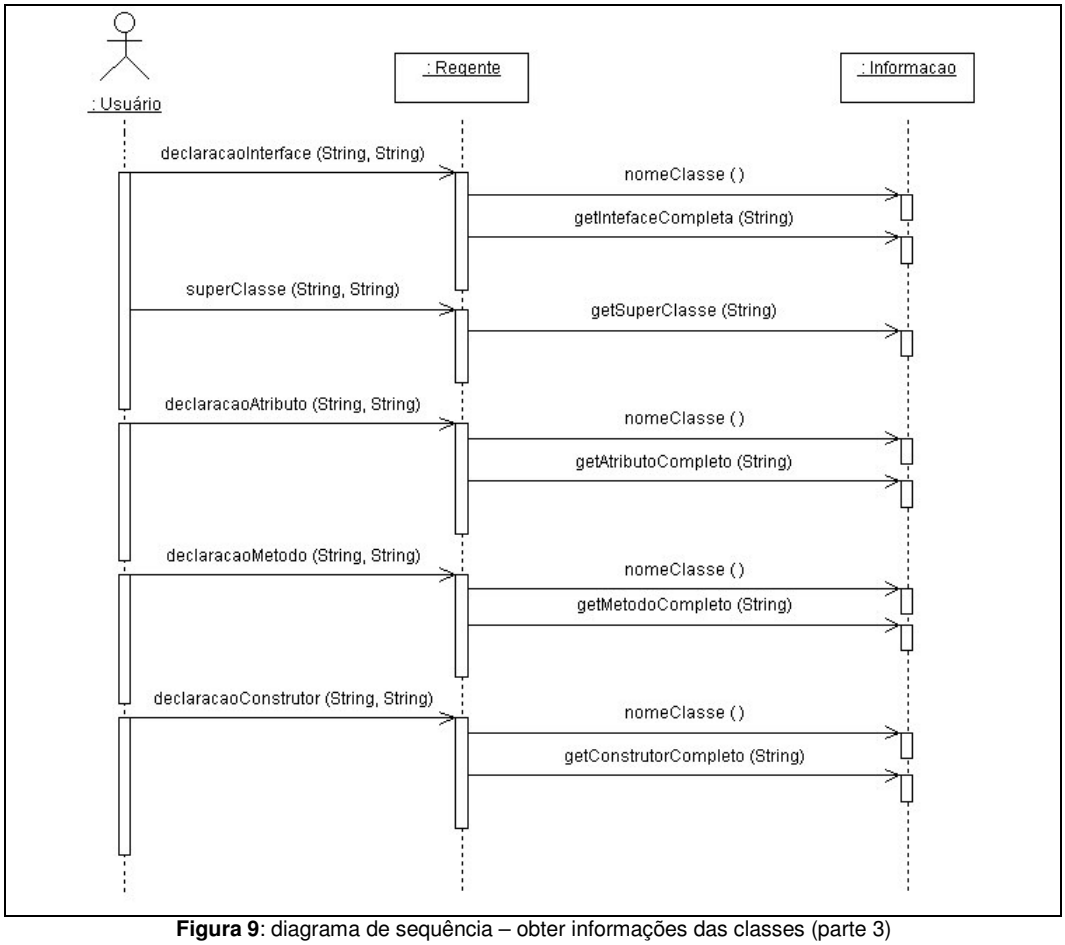


Figura 9: diagrama de sequência – obter informações das classes (parte 3)

Esta é a terceira e última parte do primeiro diagrama de seqüência. Tem-se aqui a busca pela declaração completa dos membros da classe inspecionada. Na figura 9 pode-se observar que a interface faz várias requisições ao objeto da classe Regente que as repassa aos objetos da classe Informacao.

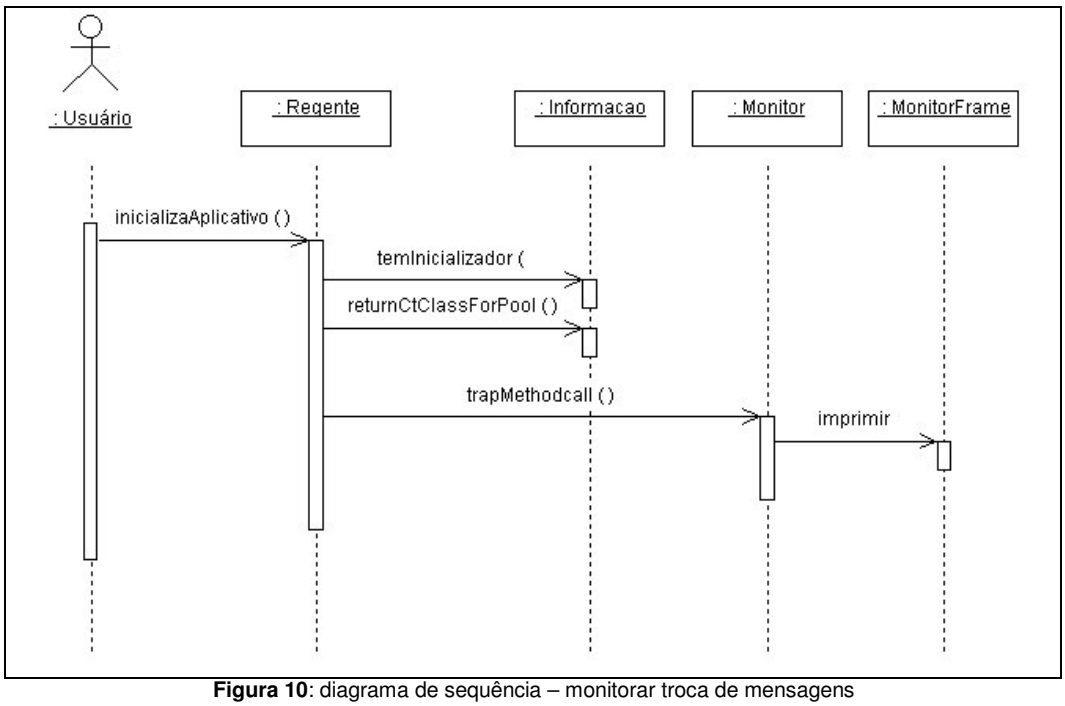


Figura 10: diagrama de sequência – monitorar troca de mensagens

O diagrama de seqüência da figura 10 trata do monitoramento do comportamento dos objetos da aplicação analisada. Para que exista o monitoramento, as classes da aplicação devem ser previamente carregadas pelo objeto de Regente, o que ocorreu no diagrama da figura 7. Em determinado momento da execução do protótipo a interface irá requisitar ao Regente a inicialização do aplicativo, como mostra o método inicializaAplicativo(). Este irá verificar junto à classe Informacao qual das classes carregadas possui o método inicializador *main* e buscará desta o *ClassPool* que possui informações referentes a ela. Após isto, pelo processo de reflexão comportamental é invocado o construtor da classe Monitor que passará a invocar o método imprimir(*String*) da classe MonitorFrame sempre que interceptar uma mudança no comportamento dos objetos do aplicativo.

A implementação do protótipo foi realizada no ambiente de programação Borland JBuilder 7 Enterprise Trial, escolhido por disponibilizar condições de desenvolvimento do protótipo proposto. Para que fosse possível utilizar os recursos do mecanismo de reflexão no JBuilder, foram realizados os seguintes procedimentos:

- durante o processo de criação da aplicação, preencheu-se o campo *Required Libraries* com o caminho até o pacote de classes Javassist;
- em cada uma das classes que usaram os benefícios das classes de reflexão da biblioteca Javassist, importou-se as classes do pacote javassist e do pacote javassist.reflect.

Para fins de demonstração, foi elaborado um aplicativo muito simples, que contivesse em seu código técnicas de orientação a objetos para que os dados de suas classes pudessem ser inspecionados e seus objetos monitorados pelo protótipo desenvolvido.

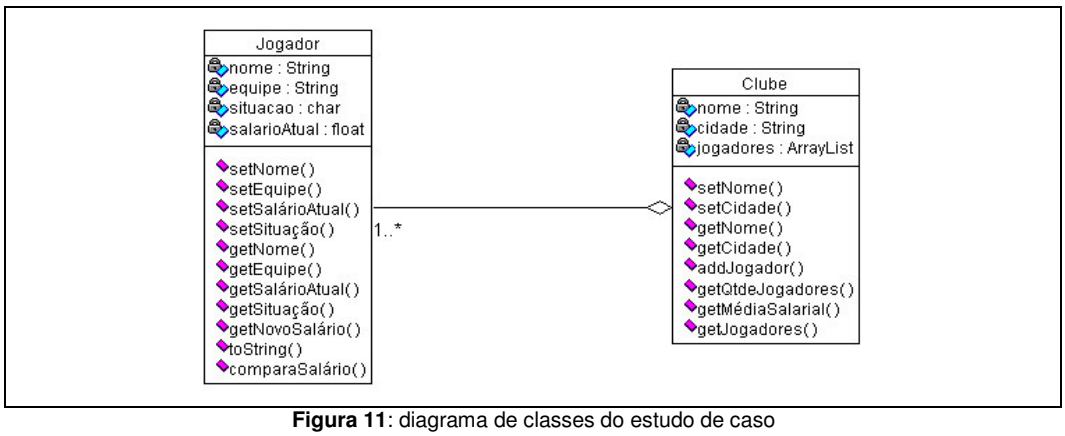


Figura 11: diagrama de classes do estudo de caso

Como demonstra a figura 12, as classes do aplicativo foram inspecionadas corretamente pelo protótipo desenvolvido. As classes apresentadas pela figura 11 estão na lista de classes do protótipo juntamente com a classe Exe4_07 que se refere à classe de interação com o usuário e por esse motivo não foi representada no diagrama de classes. Verificando-se a classe Jogador no diagrama da figura 12 encontram-se quatro atributos: nome, equipe, situacao e salarioAtual, que estão dispostos na lista de atributos da classe jogador mostrada pela figura 11. Na figura 13 estão apresentadas várias mensagens passadas aos objetos instanciados destas classes, capturadas pelo monitoramento a esta aplicação exemplo.

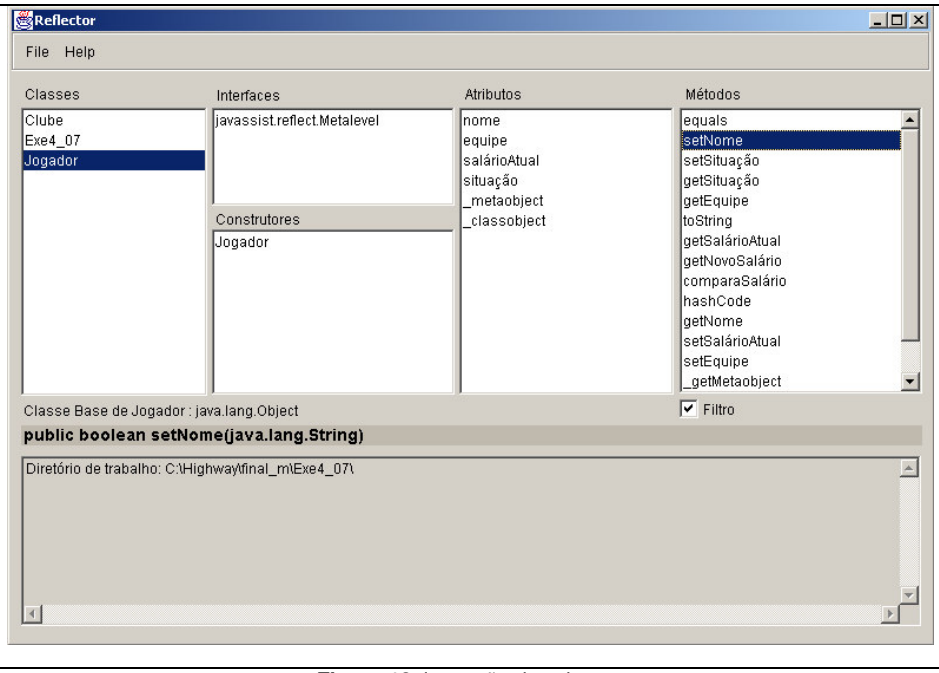


Figura 12: inspeção das classes

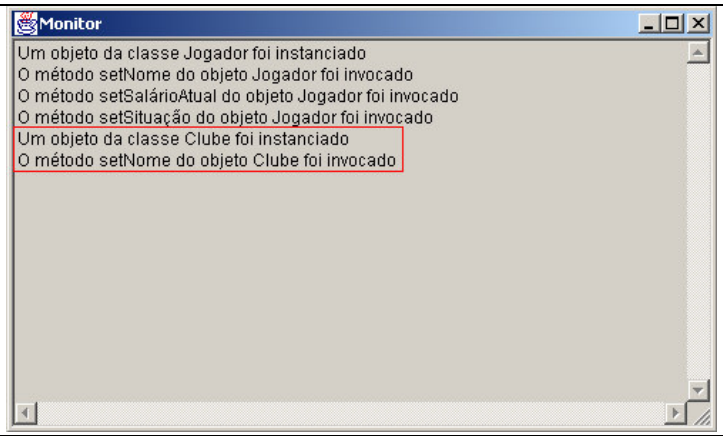

Monitor
Um objeto da classe Jogador foi instanciado
O método setNome do objeto Jogador foi invocado
O método setSalárioAtual do objeto Jogador foi invocado
O método setSituação do objeto Jogador foi invocado
Um objeto da classe Clube foi instanciado
O método setNome do objeto Clube foi invocado

Figura 13: monitoramento da aplicação exemplo

4 Conclusões

O estudo realizado por este trabalho sobre as tecnologias de reflexão computacional e orientação a objetos serviu para demonstrar a manipulação e introspecção de classes Java com o auxílio da biblioteca Javassist. Apesar da tecnologia não ser muito recente, seus conceitos e funcionalidades estão vindo à tona apenas nos últimos anos. Diversos estudos, como de Shigero Chiba (2001), vêm sendo realizados com o intuito de prover características reflexivas às plataformas em geral, sem sobrecarregar as funcionalidades, eficiência e desempenho das mesmas.

Neste trabalho utilizou-se das vantagens oferecidas pela união das tecnologias de reflexão computacional e orientação a objetos, como a possibilidade de monitoramento dos objetos. Este não visou atender completamente os conceitos de reflexão computacional apresentados, mas sim, demonstrar de forma genérica como interceptar as mensagens dos objetos, possibilitando mudanças em seu comportamento. Também não foi realizado um estudo comparativo das ferramentas disponíveis, nem seu impacto em termos de desempenho de execução.

À medida que ambientes e linguagens de programação reflexivas tornarem-se mais difundidos, é bastante provável que a utilização de reflexão computacional seja tão comum quanto a utilização de orientação a objetos. O desenvolvimento deste protótipo mostrou que as técnicas de reflexão podem ser facilmente utilizadas, mesmo no dia-a-dia do desenvolvimento de software.

Referências

BARTH, Fabrício Jailson. **Utilização de reflexão computacional para implementação de aspectos não funcionais em um gerenciador de arquivos distribuídos**. 2000. 76 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

CHIBA, Shigeru. **That are the best join points?** Tokyo, 2001. Disponível em: <http://www.csg.is.titech.ac.jp/~chiba/>. Acesso em: 01 out. 2002.

DEVEGILI, Augusto Jun. **Tutorial sobre reflexão em orientação a objetos**. Florianópolis, abr 2000. Disponível em: <http://www.uvm.edu/~dewey/reflection_manual/>. Acesso em: 23 jun. 2002.

GOLM, Michael, KLEINODER, Jurgen. **metaXa and the future of reflection**. Erlangen, jan.

1998. Disponível em: <http://www4.informatik.uni-erlangen.de/Publications/>. Acesso em: 28 ago. 2002.

KILLIJIAN, Marc-Olivier; FABRE, Jean-Charles; RUIZ-GARCIA, Juan-Carlos. **Development of a metaobject protocol for fault-tolerance using compile-time reflection**. Cedex, 1998. Disponível em: <http://www4.informatik.uni-erlangen.de/Publications/>. Acesso em: 28 ago. 2002.

SENRA, Rodrigo Dias Arruda. **Programação reflexiva sobre o protocolo de meta-objetos Guaraná,** São Paulo, nov. 2001. Disponível em: <http://www.ic.unicamp.br/~921234/dissert/node5.html>. Acesso em: 12 maio 2002.

TATSUBORI, Michaki. **OpenJava: A Class-based Macro System for Java**. Tsukuba, 1998. Disponível em: <http://www4.informatik.uni-erlangen.de/Publications/> Acesso em: 28 ago. 2002.

TATSUBORI, Michaki. **Welcome to OpenJava**. Ago. 2002. Disponível em: <http://www.hlla.is.tsukuba.ac.jp/~mich/openjava/> Acesso em: 07 ago. 2002.

WINBLAD, Ann L.; EDWARDS, Samuel D.; KING, David R. **Software orientado ao objeto**. São Paulo: Makron Books, 1993.

GRAMO: Uma Técnica para a Construção de Modelos de Domínio Reutilizáveis no Desenvolvimento de Sistemas Multiagente

Carla Gomes de Faria (UFMA)
carla@postal.elo.com.br

Rosario Girardi (UFMA)
rgirardi@deinf.ufma.br

Resumo. O Reuso de software pode aumentar a qualidade e a produtividade no desenvolvimento de aplicações. Duas abordagens complementares caracterizam o reuso de software: a Engenharia de Domínio ou desenvolvimento PARA o reuso e a Engenharia de Aplicações ou desenvolvimento COM o reuso. Neste artigo é proposta a GRAMO, uma técnica baseada em ontologias para a construção de modelos de domínio na Engenharia de Domínio multiagente. É apresentada a modelagem da ONTODM, a ontologia genérica utilizada pela GRAMO para guiar o processo de construção de modelos de domínio. A utilização da técnica é ilustrada com exemplos da área jurídica.

Palavras-chave: Análise de Requisitos, Análise de Domínio, Engenharia de Domínio Multiagente, Ontologias.

1 Introdução

O reuso de software pode aumentar a qualidade e a produtividade no desenvolvimento de aplicações. Segundo Girardi e Faria (2003) duas abordagens complementares caracterizam o reuso de software: a Engenharia de Domínio ou Desenvolvimento PARA o reuso e a Engenharia de Aplicações ou Desenvolvimento COM o reuso. A Engenharia de Domínio busca a construção de abstrações de software reutilizáveis, que serão usadas na criação de aplicações específicas na Engenharia de Aplicações.

Entre as abstrações de alto nível construídas na Engenharia de Domínio estão os modelos de domínio. Um modelo de domínio é uma abstração dependente de um domínio de aplicação particular, especificado em um alto nível de abstração, que representa a formulação de um problema, conhecimento ou atividade do mundo real. A formulação é genérica o suficiente para representar uma família de sistemas similares [GIRARDI, 2002].

Para Fouro e Werner (2001), as ontologias são estruturas de representação de conhecimento adequadas para serem utilizadas na representação de modelos de domínio, devido a características tais como, notação formal, que providencia uma terminologia clara e não ambígua; flexibilidade, que permite posteriores extensões; e a adaptabilidade em diferentes níveis de generalidade/especificidade, facilitando a reutilização.

Neste artigo é proposta a técnica GRAMO (Generic Requirement Analysis Method based on Ontologies) baseada em ontologias para a construção de modelos de domínio. Este trabalho é realizado no contexto do projeto de pesquisa MaAE (Multi-Agent Application Engineering) [GIRARDI, 2002], que busca a sistematização da Engenharia de Domínio Multiagente, através da elaboração de técnicas para a construção de abstrações reutilizáveis de alto nível: modelos de domínio, frameworks e padrões de projeto.

A técnica GRAMO é baseada na utilização de uma ontologia genérica, a ONTODM (Ontologia genérica para a Modelagem de Domínios), que representa o conhecimento de métodos para a análise de requisitos e o conhecimento de métodos da análise de domínio para construir

modelos de domínio. Na técnica GRAMO a ONTODM é utilizada para guiar o processo de construção dos modelos de domínio.

Este artigo está organizado da seguinte forma. A seção 2 mostra uma visão geral dos conceitos, das características e dos tipos de ontologias. A seção 3 apresenta a versão atual da ONTODM e o seu processo de construção. A seção 4 exibe a técnica GRAMO com exemplos ilustrativos da área jurídica.

2 Ontologias

Uma ontologia é uma especificação explícita dos objetos, conceitos e outras entidades que assumimos existirem em uma área de interesse, além das relações entre esses conceitos e restrições expressados através de axiomas [GRUBER, 1995].

Conforme Guarino (1998), as ontologias podem ser classificadas segundo o seu nível de generalidade em: ontologias genéricas, ontologias de domínio, ontologias de tarefa e ontologias de aplicação.

Uma *ontologia genérica* descreve conceitos gerais, tais como, espaço, tempo, matéria, objeto, evento, ação, sendo independentes de domínio ou problema particulares.

Uma *ontologia de domínio* reúne conceitos e seus relacionamentos em um domínio particular, definindo restrições na estrutura e conteúdo do conhecimento desse domínio; por exemplo, o domínio jurídico.

Uma *ontologia de tarefa* expressa conceitos sobre resolução de problemas, independentemente do domínio em que ocorram, isto é, descrevem o vocabulário relacionado a uma atividade ou tarefa genérica; por exemplo, o acesso à informação.

Uma *ontologia de aplicação* descreve conceitos dependentes ao mesmo tempo de um domínio particular e de um conjunto de tarefas específicas. Estes conceitos freqüentemente correspondem a papéis desempenhados por entidades do domínio enquanto realizam certas atividades; por exemplo, o acesso à informação jurídica.

Os conceitos de uma ontologia de domínio ou de uma ontologia de tarefa devem ser especializados dos termos introduzidos por uma ontologia genérica. Os conceitos de uma ontologia de aplicação, por sua vez, devem ser especializações dos termos das ontologias de tarefas e das ontologias de domínio (Figura 1). As setas expressam relacionamentos de especialização.

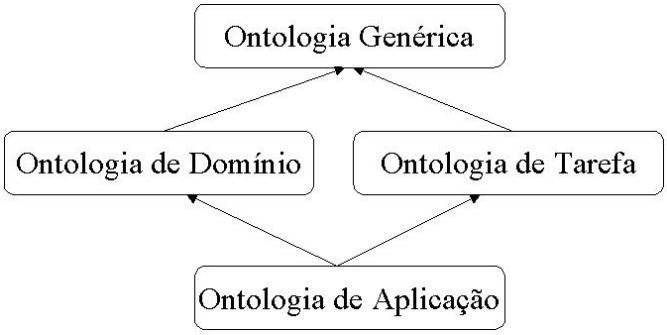


Figura 1: Classificação das ontologias segundo o seu nível de generalidade.

O uso de ontologias provê benefícios na análise e especificação de requisitos de um sistema:

- *Na comunicação:* uma ontologia é uma estrutura de representação de conhecimento útil para ajudar as pessoas a se comunicarem sobre um determinado domínio;
- *Na formalização:* a notação formal usada na especificação de um domínio utilizando ontologias elimina possíveis contradições e inconsistências, resultando, portanto, em uma especificação não ambígua;
- *Na representação do conhecimento e reuso:* uma ontologia representa um vocabulário de consenso e especifica conhecimento de domínio de forma explícita no seu mais alto nível de abstração com enorme potencial para o reuso. As ontologias providenciam uma terminologia clara reduzindo as confusões terminológicas e conceituais de termos utilizados na análise de requisitos.

3 ONTODM – Uma Ontologia para a construção de Modelos de Domínio

A ONTODM é uma ontologia genérica, que representa o conhecimento de métodos da análise de requisitos e da análise de domínio e guia a captura e especificação do conhecimento dos conceitos e das tarefas a serem realizadas no domínio.

3.1 O processo de construção da ONTODM

A figura 2 ilustra o processo de construção da ONTODM, que foi inspirado na técnica proposta por Fridman e Mcguinness (2001) e Faria e Girardi (2003) para a construção de ontologias. O processo consiste de duas fases: a definição e o projeto da ontologia. Na fase de definição é utilizado o conhecimento dos métodos da análise de requisitos e da análise de domínio para gerar uma rede semântica com a representação desses conceitos. Na fase do projeto a ONTODM (Figura 5) é criada através do mapeamento da rede semântica a uma ontologia baseada em frames representada por uma hierarquia de meta-classes.

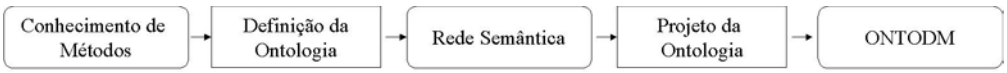


Figura 2: Processo de construção da ONTODM

3.1.1 Conhecimento de Métodos da Análise de Requisitos e da Análise de Domínio

Vários métodos para a análise de requisitos de sistemas multiagente tem sido propostos: GAIA [WOODRIDGE e CICARINI 2001], MASE [DILEO, JACOBS e DELOACH, 2002] [WOOD e DELOACH, 2001], MADS [SILVA e GIRARDI, 2002] [SODRÉ, 2002], TROPOS [CASTRO, KOLP e MYLOPOULOS, 2001] [MYLOPOULOS e CASTRO, 2000], SODA [OMICINI, 2001], PASSI [COSSENTINO *et al.*, 2002] e MESSAGE [CAIRE *et al.*, 2001]. Estes métodos, em geral são baseados nos conceitos de objetivos, papéis e interações presentes nas organizações. Não existe um consenso a respeito desses conceitos nos métodos. A seguir estão definidos esses conceitos de acordo aos critérios definidos no projeto de pesquisa MaAE.

Uma organização é composta de indivíduos com *objetivos gerais e específicos*, que estabelecem o que a organização pretende alcançar. O alcance dos *objetivos específicos* permite o alcance do *objetivo geral* da organização. Por exemplo, um sistema de informação pode ter como

objetivo geral *satisfazer as necessidades de informação de uma organização* e objetivos específicos *satisfazer as necessidades de informação pontual e satisfazer as necessidades de informação em longo prazo.*

Os *objetivos específicos* são atingidos através do cumprimento de *responsabilidades*. As *responsabilidades* são associadas a indivíduos, que desempenham *papéis* com um certo grau de autonomia. As *responsabilidades* são exercidas através da execução de *atividades*, para tanto os indivíduos fazem uso de *recursos*. Por exemplo, um indivíduo pode desempenhar o papel de *recuperador de informação* com a responsabilidade de executar atividades, que permitam *satisfazer as necessidades de informação pontuais* da organização, outro indivíduo pode desempenhar o papel de *filtrador de informação* com a responsabilidade de executar atividades, que permitam *satisfazer as necessidades de informação em longo prazo* da organização.Os *recursos* podem ser, por exemplo, *as regras da organização para acessar e estruturar suas informações.* Para a realização das *atividades*, os indivíduos necessitam *interagir* com outros indivíduos ou com *entidades externas* à organização, por exemplo, para a realização de uma *atividade entregar elementos de informação recuperados* realizada pelo *papel* de *recuperador de informação* necessita *interagir* com o papel *interfaceador com o usuário.*

De acordo com estas definições a modelagem de requisitos de sistemas multiagente é baseado na modelagem de objetivos, papéis e interações.

Na *modelagem de objetivos*, primeiro é identificado o *objetivo geral* do sistema, através do problema que o sistema se propõe a resolver. Em seguida, são identificados os *objetivos específicos* através da especialização ou refinamento do *objetivo geral*. Por último são identificadas as *responsabilidades* associadas a cada *objetivo específico* correspondente. O produto desta atividade é o *modelo de objetivo* composto de *objetivos gerais* e *específicos* e as *responsabilidades.*

Na *modelagem de papéis,* primeiro são identificadas as *atividades* baseada nas *responsabilidades* identificadas na *modelagem de objetivos*. Para cada responsabilidade é identificado um conjunto de *atividades*. É identificado um *nome* para o *papel,* que decorre da *responsabilidade* que lhe é atribuída. Em seguida, são identificados os *recursos*, que cada papel usa baseados em suas *atividades*. O produto desta atividade é o *modelo de papel* composto de *papéis* com suas respectivas *responsabilidades, atividades e recursos*.

Na *modelagem de interações,* primeiro são identificadas as *interações* entre papéis para o exercício de suas *atividades*. Durante a identificação das *interações* é feita também a identificação das *entidades externas*. O produto desta atividade é o *modelo de interações* composto de *interações* entre *papéis* e *entidades externas*.

A Análise de Domínio modela os conceitos e funcionalidades requeridas por uma família de sistemas [WERNER e BRAGA, 2000]. Os métodos para a Análise de Domínio como a abordagem de Prieto Diaz [PRIETO-DIAZ, 1987], o método Shaller e Mellor [SHLAER e MELLOR, 1989] e o método FODA (Feature Oriented Domain Analyse) [COHEN *et al.*, 1990] [KRUT, 1993] em geral indicam como representar os conceitos de domínio e os relacionamentos existentes entre os conceitos.

A *modelagem de conceitos* relevantes do domínio é feita através de uma análise das fontes de informação relevantes no domínio como: livros, revistas, artigos, relatórios e os especialistas no domínio podem ser consultados. Em seguida é feita uma análise de aplicações existentes no referido domínio e é coletado o que existe em comum nessas aplicações. Segundo essas análises, inicia-se a captura dos conceitos de domínio, a descrição textual desses conceitos e o estabelecimento de relacionamentos, que são representadas pelo produto *modelo de conceitos*.

3.1.2 Definição da Ontologia

Nesta fase o conhecimento dos métodos da análise de requisitos e da análise de domínio é representado em uma rede semântica (Figura 3 e Figura 4).

A Figura 3 mostra a parte da rede semântica representando o conhecimento dos métodos da análise de requisitos e da análise de domínio relacionado aos conceitos da modelagem, seus relacionamentos e atributos: o objetivo, a responsabilidade, a atividade, o conceito do domínio, o relacionamento entre os conceitos do domínio, o recurso, a interação que ocorre entre o papel e a entidade externa ou entre papéis. Objetivo é classificado em geral ou específico. Os objetivos específicos conduzem ao objetivo geral e são alcançados através do exercício da responsabilidade do papel, por meio da realização de atividades. Os papéis usam os recursos para a realizar suas atividades. As atividades podem manipular os conceitos do domínio. As interações têm fontes e destinos, que podem ser os papéis ou entidades externas. Os conceitos do domínio podem estabelecer relacionamentos particulares.

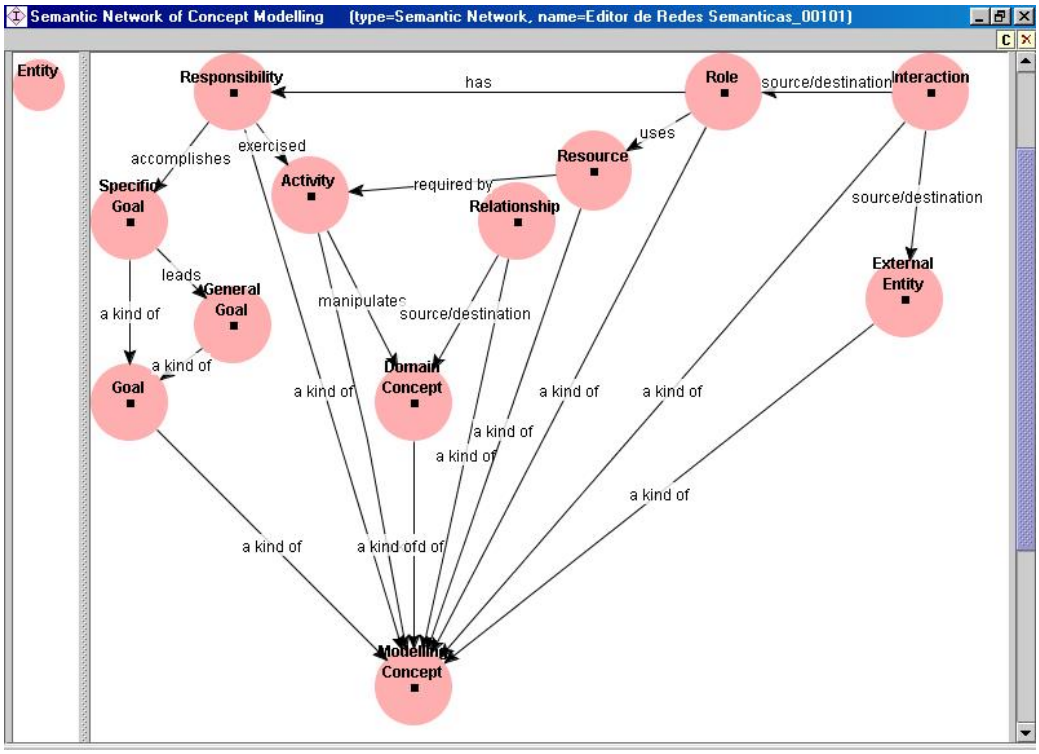


Figura 3: Rede Semântica dos Conceitos da Modelagem da ONTODM

A Figura 4 mostra parte da rede semântica representando o conhecimento dos métodos da análise de requisitos e da análise de domínio relacionados às tarefas da modelagem e os produtos gerados. A modelagem de domínio constrói o modelo de domínio composto dos modelos de conceitos, de objetivo, de papel e de interações, que são construídos através das subtarefas de modelagem de conceitos, de objetivos, de papéis e de interações respectivamente. A modelagem de conceitos constrói o modelo de conceitos representando os conceitos do domínio e os relacionamentos existentes entre eles. A modelagem de objetivos constrói o modelo de objetivo

representando os objetivos gerais e específicos e as responsabilidades. A modelagem de papéis constrói o modelo de papel representando os papéis. A modelagem de interações constrói o modelo de interações representando os papéis, as entidades externas e as interações existentes entre eles. Os modelos de domínio, de conceitos, de objetivo, de papel e de interações são os produtos gerados.

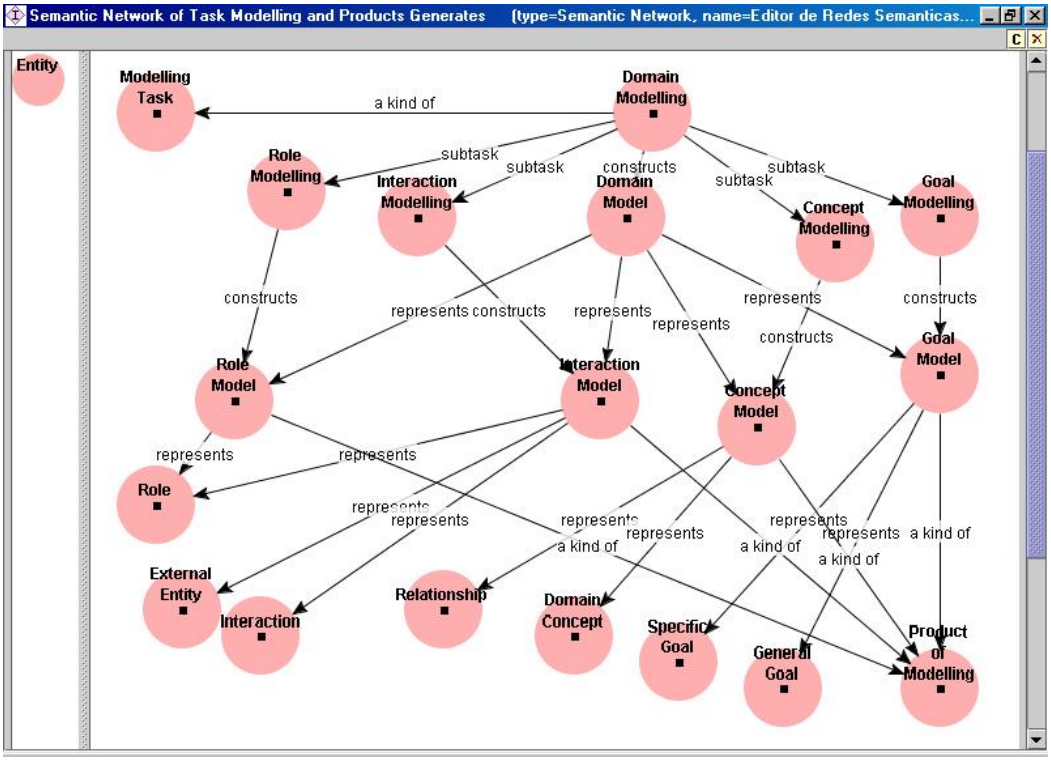


Figura 4: Rede Semântica das Tarefas e Produtos da Modelagem da ONTODM

3.1.3 Projeto da Ontologia

Na fase do projeto da ontologia os conceitos e relacionamentos da rede semântica são mapeados à ONTODM. Os nós são mapeados para meta-classes. Os nós relacionados por um link "a kind of" são mapeados em uma hierarquia de subclasses e superclasses. Outros links são mapeados para slots correspondentes a meta-classes. Cada slot é associado com facetas apropriadas como tipo e cardinalidade.

Na Figura 5 são visualizados os slots da meta-classe papel e a hierarquia taxonômica da ONTODM.

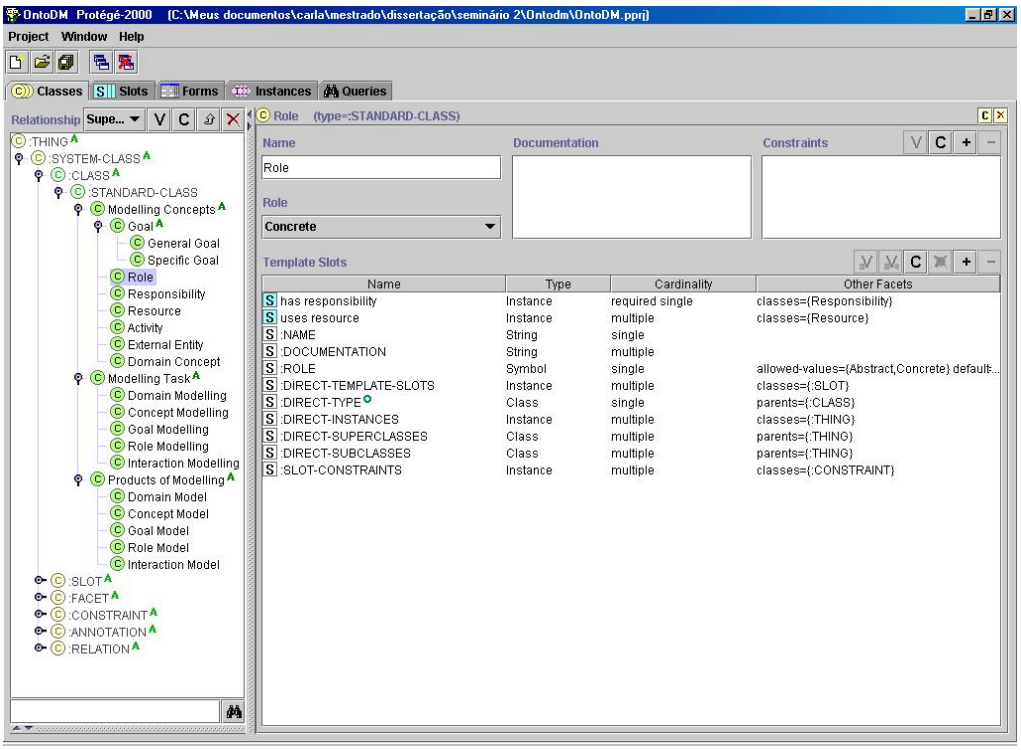


Figura 5: Hierarquia de meta-classes da ONTODM e a meta-classe papel

4 GRAMO – Uma técnica baseada em Ontologias para a Modelagem de Domínios na Engenharia de Aplicações

A técnica GRAMO (Figura 6) define as atividades a serem realizadas na construção de modelos de domínio, através da instanciação da ONTODM segundo os conceitos e as tarefas do domínio.

Figura 6: Os insumos e os produtos da GRAMO

Através da instanciação da meta-classe Domain Modelling será criada a classe Domain Model contendo a especificação dos conceitos e tarefas no domínio. A atividade de modelagem de domínio é decomposta nas subtarefas modelagem de conceitos, modelagem de objetivos, modelagem de papéis e modelagem de interações, que geram os produtos modelo de conceitos, modelo de objetivo, modelo de papel e modelo de interações. A seguir serão descritos os processos de modelagens com exemplos ilustrativos referentes à área jurídica (REALE, 1999) (CINTRA, GRINOVER e DINAMARCO, 1999).

Na modelagem de conceitos são criadas instâncias dos conceitos do domínio, por exemplo, ***conduta, fato, fonte formal, fonte material, função imperativa, função legitimadora, função modificadora, função processual, norma, princípio, regra, regra de conduta, regra de organização, sanção e valor***. Em seguida, são criadas instâncias dos relacionamentos existentes entre esses conceitos do domínio, por exemplo, ***fonte material e fonte formal são tipos de fonte; fonte material inspira norma; fonte formal exterioriza norma; regra e princípio são tipos de norma; regra de conduta e regra de organização são tipos de regra; regra de conduta suporta regra de organização; regra de conduta possui função legitimadora, função processual, função imperativa e função modificadora; regra de organização prescreve sanção; regra de organização indica conduta; regra de organização prevê fato; conduta evita sanção; conduta decorre do fato; valor dá significado ao fato e valor orienta a aplicação da norma***. Os conceitos e os relacionamentos são identificados através de uma análise das fontes de informação relevantes no domínio como: livros, revistas, artigos, relatórios e os especialistas no domínio; e das aplicações existentes no referido domínio é coletado o que existe em comum nessas aplicações. A Figura 7 mostra o modelo de conceitos da área jurídica.

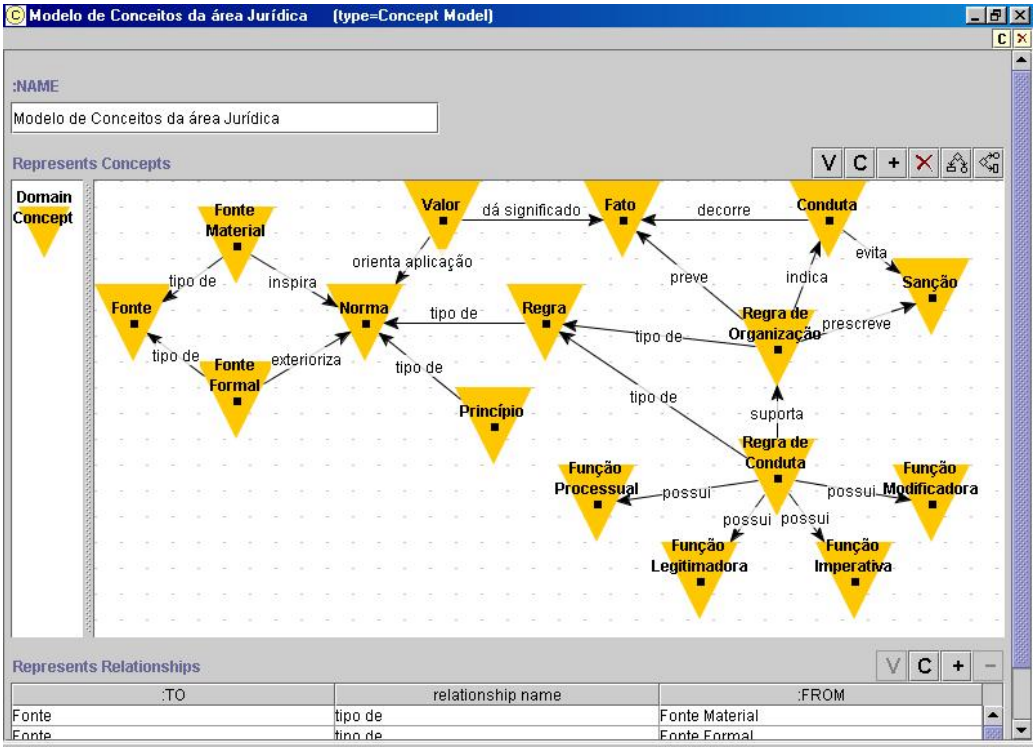


Figura 7: Modelo de Conceitos da área Jurídica

Na modelagem de objetivos são criadas instâncias dos objetivos gerais e específicos, por exemplo, ***objetivo geral: harmonizar as relações sociais com justiça e objetivos específicos: pacificar conflitos entre indivíduos (social), preservar o ordenamento jurídico (político) e promover a efetiva aplicação das normas (jurídico)***. Em seguida, os objetivos específicos são vinculados ao objetivo geral. Por último são criadas instâncias das responsabilidades e vinculadas aos objetivos específicos correspondentes, por exemplo, ***as responsabilidades apresentação dos***

fatos que fundamentam o litígio, contestação dos fatos que fundamentam o litígio e apreciação dos fatos e interpretação das normas são associadas ao objetivo específico promover a efetiva aplicação das normas (jurídico). O objetivo geral é identificado através do problema que o sistema se propõe a resolver. Os objetivos específicos são identificados através do refinamento ou especialização do objetivo geral. As responsabilidades são identificadas a partir dos objetivos específicos, ou seja, as responsabilidades levam ao cumprimento dos objetivos específicos. A Figura 8 mostra o modelo de objetivo da área jurídica.

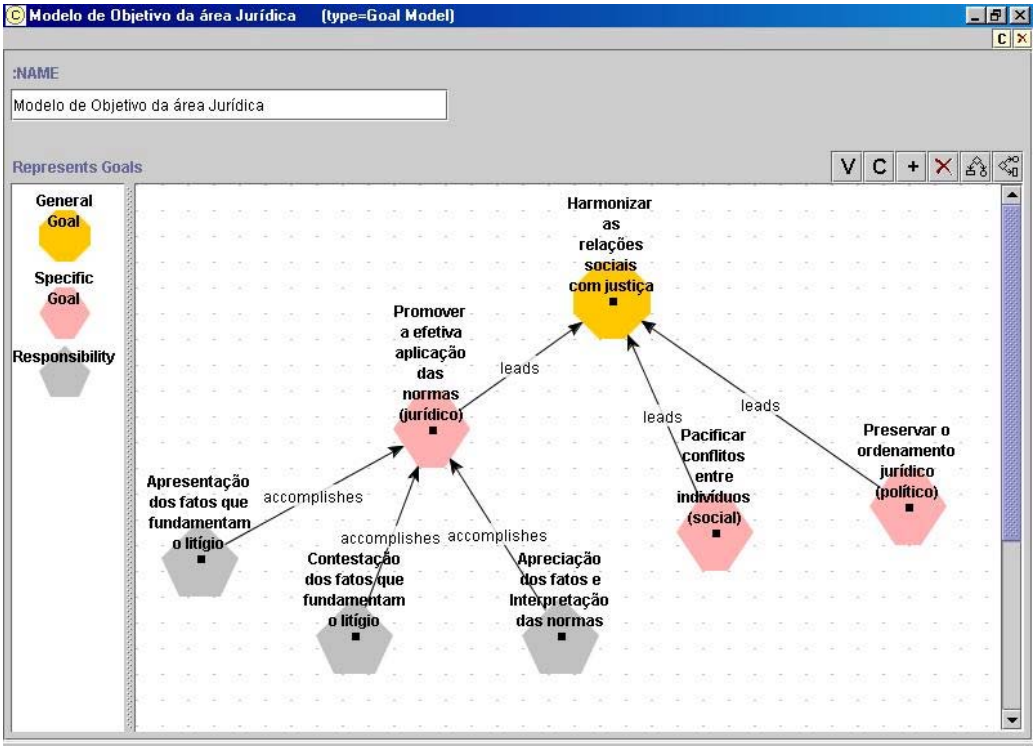


Figura 8: Modelo de Objetivo da área Jurídica

Na modelagem de papéis são criadas instâncias do papel, por exemplo, ***autor, juiz e réu***. Em seguida, são adicionadas responsabilidades e vinculadas ao papel, por exemplo, ***a responsabilidade apresentação dos fatos que fundamentam o litígio é associada ao papel autor; a responsabilidade contestação dos fatos que fundamentam o litígio é associada ao papel réu; e a responsabilidade apreciação dos fatos e interpretação das normas é associada ao papel juiz***. São criadas instâncias das atividades e vinculadas às responsabilidades, por exemplo, ***as atividades apresentar provas e apresentar demanda são vinculadas à responsabilidade apresentação dos fatos que fundamentam o litígio; a atividade apresentar defesa é vinculada à responsabilidade apresentação dos fatos que fundamentam o litígio; e as atividades ordenar produção de provas e decidir causa são vinculadas à responsabilidade apreciação dos fatos e interpretação das normas***. São adicionados os conceitos do domínio e vinculadas às atividades. Por último, são criadas instâncias dos recursos e vinculados às atividades, por exemplo, ***os recursos constituição, códigos e leis são vinculados a atividade decidir causa; o recurso provas é vinculado a atividade apresentar provas***. As atividades são identificadas à partir das responsabilidades, ou seja, uma

responsabilidade é exercida através de um conjunto de atividades. As atividades manipulam conceitos do domínio. Os recursos são identificados baseados nas atividades, isto é, os recursos são requeridos pelas atividades. A Figura 9 mostra o modelo de papel da área jurídica.

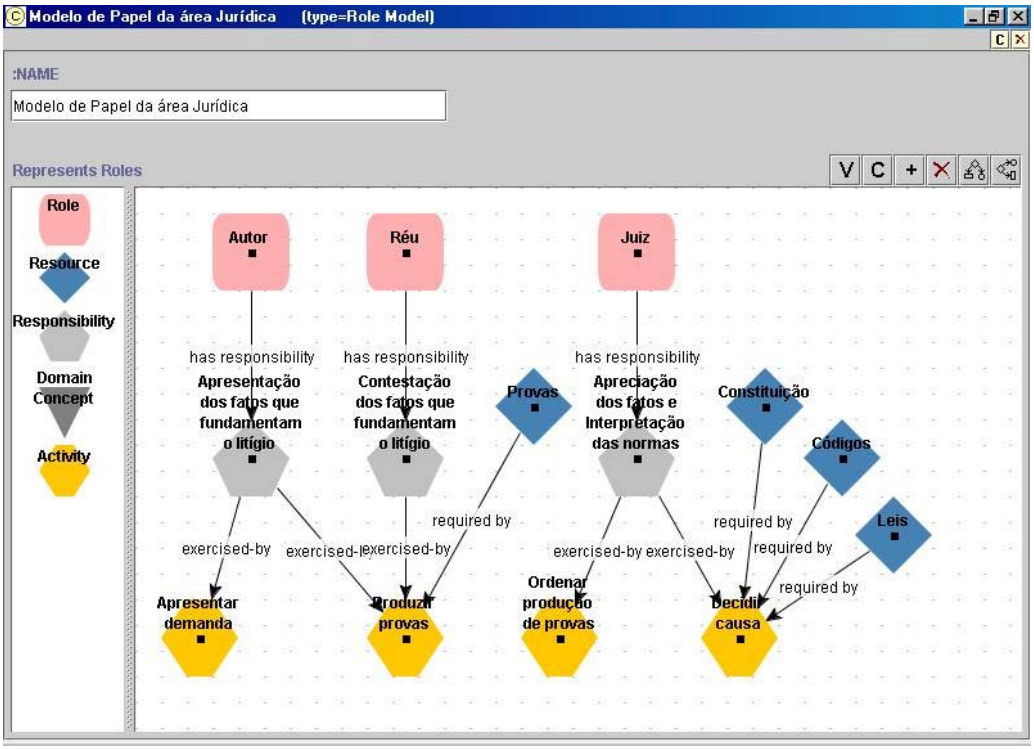
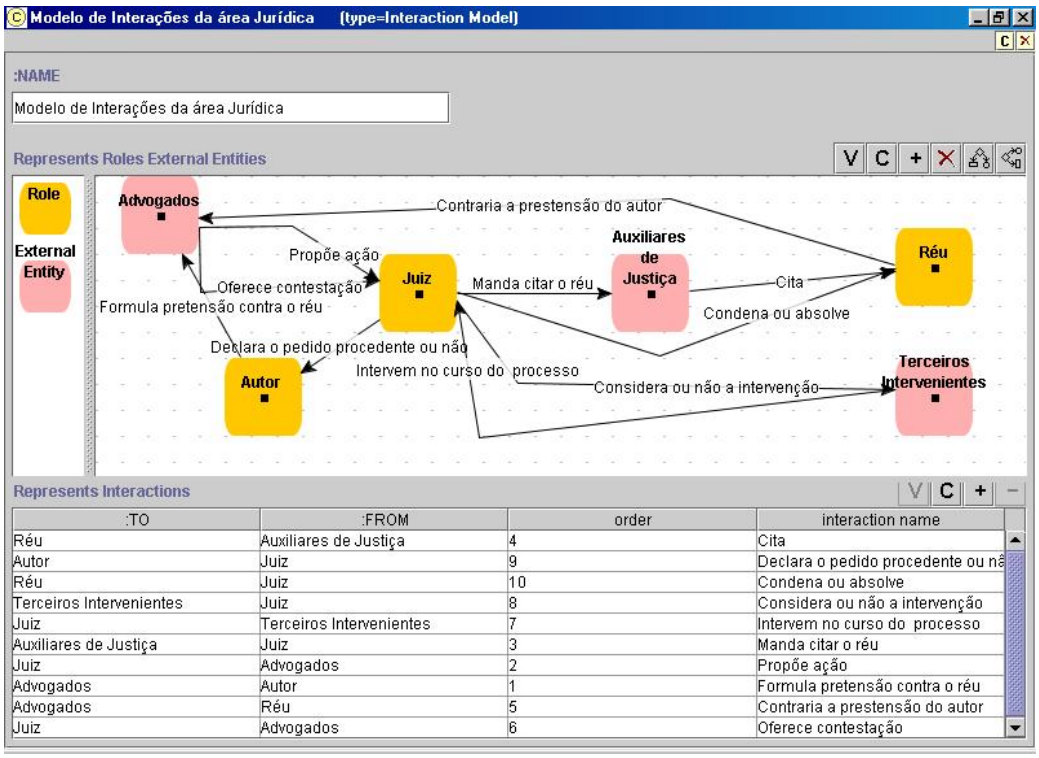


Figura 9: Modelo de Papel da área Jurídica

Na modelagem de interações são criadas instâncias das entidades externas, por exemplo, ***terceiros intervenientes, advogados e auxiliares de justiça***. Em seguida, são adicionados os papéis, ***juiz, autor e réu***. Por último, são criadas as interações entre os papéis e entidades externas, por exemplo, ***formula pretensão contra o réu (ordem: 1, origem: autor e destino: advogado); propõe ação (ordem: 2, origem: advogado e destino: juiz); manda citar o réu (ordem: 3, origem: juiz e destino: auxiliar de justiça); cita (ordem: 4, origem: auxiliar de justiça e destino: réu); contraria pretensão do autor (ordem: 5, origem: réu e destino: advogado); oferece contestação (ordem: 6, origem: advogado e destino: juiz); intervem no curso do processo (ordem: 7, origem: terceiros intervenientes e destino: juiz); considera ou não a intervenção (ordem: 8, origem: juiz e destino: terceiros intervenientes); declara o pedido procedente ou não (ordem: 9, origem: juiz e destino: autor); condena ou absolve (ordem: 10, origem: juiz e destino: réu);***. As interações e as entidades externas são identificadas baseada nas atividades que os papéis realizam. A Figura 10 mostra o modelo de interações da área jurídica. A Figura 11 dá uma visão geral do exemplo do modelo de domínio para área jurídica.



:TO	:FROM	order	interaction name
Réu	Auxiliares de Justiça	4	Cita
Autor	Juiz	9	Declara o pedido procedente ou nã
Réu	Juiz	10	Condena ou absolve
Terceiros Intervenientes	Juiz	8	Considera ou não a intervenção
Juiz	Terceiros Intervenientes	7	Intervem no curso do processo
Auxiliares de Justiça	Juiz	3	Manda citar o réu
Juiz	Advogados	2	Propõe ação
Advogados	Autor	1	Formula pretensão contra o réu
Advogados	Réu	5	Contraria a prestensão do autor
Juiz	Advogados	6	Oferece contestação

Figura 10: Modelo de Interações da área Jurídica

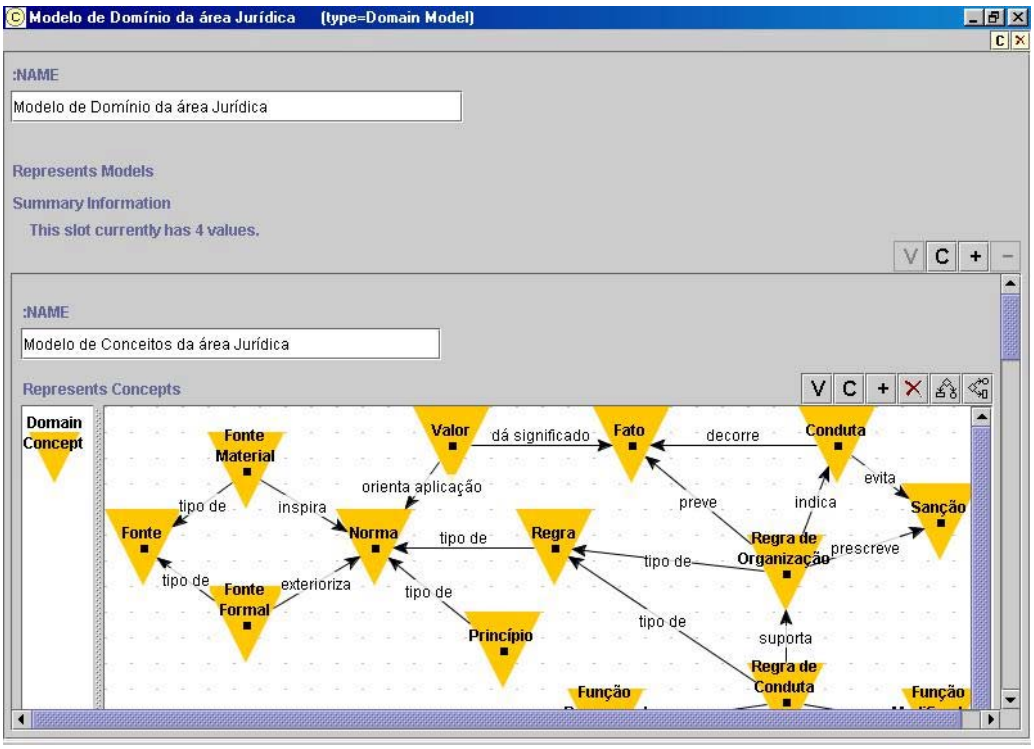


Figura 11: Visão geral do Modelo de Domínio da área Jurídica

5 Conclusões

Este artigo propõe a técnica GRAMO para a construção de modelos de domínio baseados em ontologias. A técnica propõe uma nova abordagem do desenvolvimento de software PARA o reuso utilizando a ONTODM, uma ontologia genérica que guia o processo de construção de modelos de domínio. A validação da técnica está sendo feita através da construção de modelos de domínio nas áreas do acesso à informação jurídica e turística. A ONTODM e os modelos de domínio construídos estão sendo usados na elaboração de uma técnica baseada em padrões e ontologias para a construção de frameworks multiagente.

O objetivo final é a construção de uma metodologia para a Engenharia de Domínio Multiagente, que aborde todas as fases de desenvolvimento de aplicações de software baseada em agentes.

Como trabalho futuro será feita uma extensão da ONTODM, para guiar a construção de modelos de usuário e da GRAMO para suportar a modelagem de usuários.

Agradecimento

Este trabalho é apoiado pelo CNPq.

Referências

CAIRE, G. *et al.* **"Agent-Oriented Analysis using MESSAGE/UML"**, In M. Woodridge, P. Ciancarini, and G. Weirs, editors. Second International Workshop on Agent-Oriented Software Engineering, AOSE 2001, p 101-108, 2001.

CASTRO, Joelson; KOLP, Manuel; MYLOPOLUS, John. **"A Requirement-Driven Software Development Methodology"**, 13th International Conference on Advanced Information Systems Engineering CAISE01, Interlaken, Switzerland, 4-8 June, 2001.

CINTRA, Antonio Carlos de Araújo; GRINOVER, Ada Pellegrini; DINAMARCO, Cândido R.. **"Teoria geral do processo"**. 15 ed. São Paulo: Malheiros Editores, 1999.

COHEN, S. *et al.* **"Feature Oriented Domain Analysis (FODA) Feasibility Study"**, Techinical Report CMU/SEI-90-TR-21. Software Engineering Institute, Cornegie Mellon University, Pittsburgh, PA, November, 1990.

COSSENTINO, M. *et al.* **"Introducing Pattern Reuse in the Design of Multi-agent Systems"**, AITA02, Workshop at NODE 02, p 8-9, Efurt, Germany, October, 2002.

DILEO, Jonathan; JACOBS, Timothy; DELOACH, Scott. **"Integrating Ontologies into Multiagent Systems Engineering"**, 4th International Bi-Conference Workshop on Agent Oriented Information Systems (AOIS 2002), Bologna (Italy), 15-16 July, 2002.

FARIA, Carla; GIRARDI, Rosario. **"Especificação de uma Ontologia Genérica para a Análise de Requisitos da Engenharia de Aplicações Multiagente",** Terceiro Congresso Brasileiro de Computação (CBComp 2003), UNIVALI, Itajaí, SC, Brasil. Agosto de 2003. A ser publicado.

FOURO, A. M. M.; WERNER, C. M. L. **"Modelos de Domínio ou Ontologias?"**, RTInfo – Revista da Tecnologia da Informação, 2001.

FRIDMAN, N. N.; MCGUINNESS, D. L**. "Ontology Development 101: A Guide to Creating Your First Ontology"**, Knowledge Systems Laboratory, March, 2001.

GIRARDI, Rosario; FARIA, Carla. **"A Generic Ontology for the Construction of Domain Models"**, artigo submetido, 2003.

GIRARDI, Rosario. **"Reuse in Agent-based Application Development"**, In: Proceedings of 1º International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS'2002), May, 2002.

GRUBER, T. R. **"Toward Principles for the Design of Ontologies used for Knowledge Sharing"**, International Journal of Human-Computer Studies. Nº 43, pp 907-928, 1995.

GUARINO, Nicola. **"Formal Ontology in Information Systems"**, Proceedings of FOIS'98, Trento, Italy. Amsterdan, IOS Press, pp. 3-15, 6-8 June, 1998.

KRUT, R. **"Integrating 001 Tool Support into the Feature Oriented Domain Analysis Methodology"**, Technical Report CMU/SEI-93-TR-11. Software Engineering Institute, Carnegie Mellon University, 1993.

MYLOPOULOS, Jonh; CASTRO, Joelson. **"TROPOS: A Framework for Requirements-Driven Software Development"**, In J. Brinkkemper and A. Solvberg (edts), Information Systems Engineering: State of the Art and Research Themes, Lecture Notes in Computer Science, Springer, Verlag, p 261- 273, June, 2000.

OMICINI, Andrea. **"SODA Societies and Infrastructures in the Analysis and Design of Agent-based Systems"**, First International Workshop, AOSE 2000 on Agent-Oriented

Software Engineering, p 185-193, Limerick, Ireland, January, 2001.

PRIETO-DÍAZ, R. **"Classifying Software for Reuse"**, IEEE Software, Vol. 4, nº01, January, 1987.

REALE, Miguel. **"Lições preliminares de direito"**. 24 ed. São Paulo: Editora Saraiva, 1999.

SHLAER, S.; MELLOR, S**. "An Object-Oriented Approach to Domain Analysis"**, ACM SIGSOFT Software Engineering Notes, Vol. 14, Nº 5, pp. 66-77, July, 1989.

SILVA, José Henrique Alves da; GIRARDI, María Del Rosário. **"SIMCAP: Um Sistema Multiagente para a Captura de Publicações Científicas na Web"**, Revista Eletrônica de Iniciação Científica da SBC, Março, 2002.

SODRÉ, Alídia Clícia Silva. **"MADS: Uma Metodologia para o Desenvolvimento de Sistemas Baseados em Agentes"**, Conferência Ibero-americana em Sistemas, Cibernética e Informática (CISCI 2002), Julho, 2002.

WERNER, C. M. L.; BRAGA, R. M. M. **"Desenvolvimento Baseado em Componentes"**, XIV Simpósio Brasileiro de Engenharia de Software, Mini-curso, João Pessoa, Outubro, 2000.

WOOD, M. F.; DELOACH, S. A. **"An Overview of the Multiagent Systems Engineering Methodology"**, In Agent-Oriented Software Engineering - Proceedings of the First International Workshop on Agent-Oriented Software Engineering, 10th June 2000, Limerick, Ireland. P Cicarini, M. Woodridge, editors. Lecture Notes in Computer Science. Vol. 1957, Springer Verlag, Berlin, January, 2001.

WOODRIDGE, M.; CICARINI, P**. "Agent-Oriented Software Engineering: the State of the Art"**, In P. Cicarini and M. Woodridge, editors, Agent Oriented Software Engineering, Springer, Verlag, 2001.

Padrões baseados em Agentes para a Modelagem de Usuários

Ismênia Ribeiro de Oliveira (UFMA)
ism_oliveira@yahoo.com.br

Rosario Girardi (UFMA)
rgirardi@deinf.ufma.br

Resumo: A maioria dos sistemas de software não tem a habilidade para satisfazer as necessidades heterogêneas dos seus usuários. Cada usuário tem níveis de conhecimento, necessidades, habilidades e preferências bastante variadas. Nesse contexto surge a necessidade de construir sistemas que se adaptem a cada tipo de usuário ou grupo de usuários com características comuns. Este artigo aborda a modelagem de usuário como forma de produzir efeitos de adaptação em um sistema e propõe soluções recorrentes para problemas recorrentes de projetos de modelagem de usuários na forma de padrões arquiteturais e de projeto.

Palavras-chave: Modelagem de usuários, Sistemas Multiagente, Padrões de Software.

1 Introdução

A maioria dos sistemas de software não tem a habilidade para satisfazer as necessidades heterogêneas dos seus usuários. Cada usuário tem níveis de conhecimento, habilidades e preferências bastante variadas. Nesse contexto surge a necessidade de construir sistemas que se adaptem a cada tipo de usuário ou grupo de usuários com características comuns.

É possível adaptar sistemas de acordo com o conhecimento ou experiência do usuário, histórico de ações anteriores, propriedades cognitivas (estilo de aprendizado, personalidade), objetivos e planos (intenções) do usuário, seus interesses e preferências.

Um importante componente dos sistemas interativos adaptáveis é a habilidade para modelar os usuários do sistema. A modelagem de usuários é essencial em sistemas que tentam adaptar seu comportamento aos usuários para interagir de forma mais inteligente e individualizada.

Um grande desafio encontrado na modelagem de usuários é tratar a heterogeneidade dos usuários de software. Uma solução para este problema seria projetar e implementar a modelagem dos usuários e a adaptabilidade dos sistemas através de utilização da abordagem de agentes. Seriam usados agentes especializados e individualizados para cada usuário ou grupos de usuários, como agentes de interface, agentes de modelagem e agentes de adaptação.

Um agente é uma entidade autônoma, que possui um sistema interno de tomada de decisões, agindo sobre o mundo e sobre outros agentes que o rodeiam e por fim, que é capaz de funcionar sem a necessidade de algo ou alguém para guiá-lo. A analogia feita com agentes no mundo real nos leva a conceituar um agente como uma entidade ativa, sempre ao lado do usuário e que possui conhecimentos específicos sobre um determinado domínio. De posse de bases de conhecimento e de mecanismos de raciocínio, os agentes devem ser capazes de reconhecer situações em que devem se ativar, sem que o usuário perceba, ou seja, de forma transparente ao usuário [WOOLDRIDGE e JENNINGS, 1994].

Segundo Wooldridge e Jennings [WOOLDRIDGE e JENNINGS, 1994], a construção e a especificação da estrutura e funcionamento de um agente genérico pode ser realizada segundo três tipos de arquiteturas:

Arquiteturas deliberativas: segue a abordagem clássica da Inteligência Artificial, onde os agentes contêm um modelo simbólico do mundo, explicitamente representado, e cujas decisões (ações) são tomadas via raciocínio lógico, baseadas em casamento de padrões e manipulações simbólicas;

Arquiteturas reativas: a arquitetura reativa é aquela que não inclui nenhum tipo de modelo central e simbólico do mundo e não utiliza raciocínio complexo e simbólico. Baseia-se na proposta de que um agente pode desenvolver inteligência a partir de interações com seu ambiente, não necessitando de um modelo pré-estabelecido;

Arquiteturas híbridas: a arquitetura híbrida mistura componentes das arquiteturas deliberativas e reativas com o objetivo de torná-la mais adequada e funcional para a construção de agentes.

Essas considerações feitas sobre agentes nos levam a crer que a modelagem de usuários seria tremendamente melhorada e facilitada caso fosse implementada com agentes inteligentes, além da melhora na eficiência e eficácia do sistema, através dos mecanismos de adaptação.

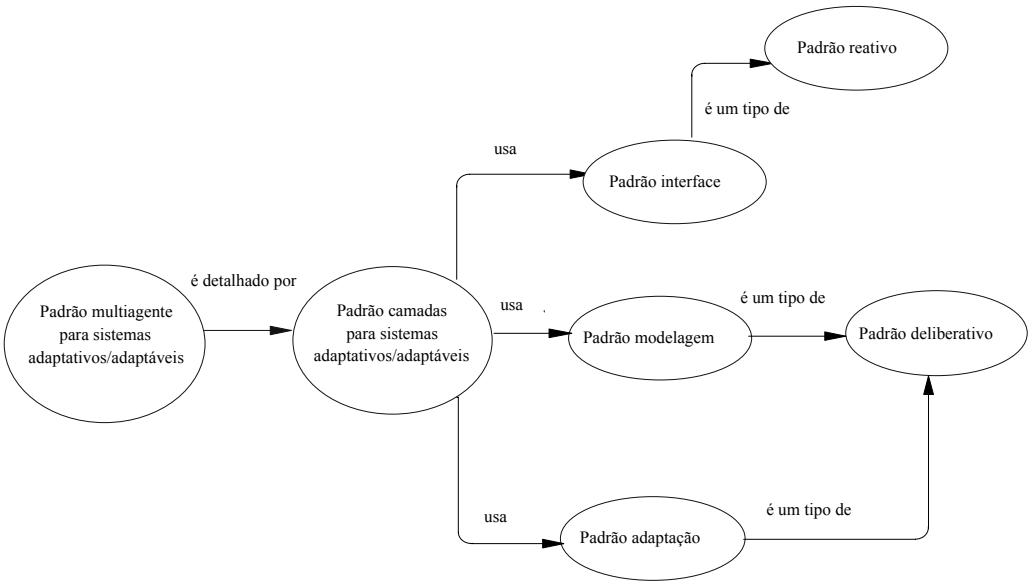


Figura 1 Rede semântica mostrando o relacionamento entre os padrões propostos

Este artigo aborda a modelagem de usuário como forma de produzir efeitos de adaptação em um sistema. A partir desse estudo são propostas soluções recorrentes para problemas recorrentes de projetos de modelagem de usuário, ou seja, padrões arquiteturais e de projeto especializados na modelagem de usuário e adaptação de sistemas. A **Figura 1** mostra, em uma rede semântica, o relacionamento entre os padrões para esses sistemas. O padrão multiagente para sistemas adaptativos/adaptáveis justifica a escolha da abordagem de agentes para sistemas adaptativos. A organização dos agentes no sistema multiagente é descrita pelo padrão arquitetural camadas multiagente para sistemas adaptativos/adaptáveis que distribui os agentes em camadas conforme as tarefas que serão executadas pelos mesmos. O detalhamento dos agentes que compõem as camadas é descrito pelos padrões interface, modelagem e adaptação. Os padrões modelagem e adaptação, por sua vez, são derivados do padrão deliberativo. Já o padrão interface é derivado do padrão reativo.

Neste artigo serão apresentados os padrões: camadas multiagente para sistemas adaptativos/adaptáveis, deliberativo, e modelagem: A **Tabela 1** apresenta um resumo desses padrões, descrevendo o problema e a solução do padrão.

Tabela 1 Padrões para o projeto de sistemas adaptativos/adaptáveis

Problema	Solução	Padrão
Como estruturar uma comunidade de agentes heterogêneos para construir modelos de usuários e se adaptar às necessidades de diferentes tipos de usuários ou grupos de usuários?	O padrão agrupa os agentes em três camadas, de acordo com as principais funcionalidades exercidas por sistemas adaptativos/adaptáveis: camada de interface; que colhe informações sobre usuários; camada de modelagem, que cria e mantém modelos de usuários; camada de adaptação, que cria modelos de adaptação.	Camadas Multiagente para Sistemas Adaptativos/Adaptáveis
Como projetar um agente para que possa raciocinar sobre um problema de forma que o possibilite atingir metas pró-ativamente dentro do contexto no qual está inserido?	O agente deve ser do tipo deliberativo, ou seja, ele deve possuir modelos simbólicos internos de do ambiente no qual ele está inserido e de si mesmo e raciocinar sobre esses modelos para criar um plano para atingir suas metas. A solução do padrão estrutura o agente em cinco camadas: camada de comunicação, que envia e processa mensagens;camada de ação, que atua no ambiente executando ações; camada de raciocínio, onde o agente elabora metas e planos; camada sensitiva, através da qual o agente percebe do ambiente; camada de conhecimento, onde o agente armazena informações sobre o estado corrente do ambiente	Deliberativo
Como criar modelos de usuários para avaliar comportamentos e conhecimentos desses usuários para que possam ser utilizados na adaptação de sistemas multiagente?	A solução envolve a criação de um agente de modelagem do tipo deliberativo, cuja estrutura é descrita pelo padrão deliberativo, adicionando nessa estrutura conhecimento e ações específicos para que o agente seja capaz de executar a tarefa modelagem de perfis de usuários.	Modelagem

Este artigo está organizado da seguinte forma: na seção 2 são descritos os principais conceitos sobre padrões de software; na seção 3 é feito um apanhado geral sobre os conceitos da modelagem de usuários e sistemas adaptativos/adaptáveis; na seção 4 é descrito o padrão camadas multiagente para sistemas adaptativos/adaptáveis, o padrão deliberativo e o padrão modelagem; a conclusão e trabalhos futuros são apresentados na seção 5.

2 Padrões de Software

As técnicas de reuso de software têm contribuído de forma significativa para reduzir o tempo e o custo do desenvolvimento de software. Uma abordagem que está sendo bastante disseminada é o uso de padrões de software para promover reutilização [GIRARDI, 2002].

Em particular, os padrões de software são necessários para o desenvolvimento de sistemas multiagente, pois eles constituem-se de soluções melhoradas para problemas recorrentes e podem ser aplicados durante o desenvolvimento de aplicações multiagente em problemas conceituais, arquiteturais ou de projeto [GIRARDI, 2001].

Um padrão apresenta uma solução bem sucedida a um problema recorrente. Ele mostra não só a solução, como também as suas restrições e também o contexto em que se deve aplicar essa solução. Um padrão possui uma forma ou estrutura identificável e reconhecida em diferentes domínios. Os padrões ajudam a diminuir a complexidade do software em várias fases do seu ciclo de vida.

Um padrão é descrito por vários atributos que podem variar de acordo com os critérios de descrição adotados. Porém, existem atributos essenciais que devem ser claramente identificáveis ao se ler um padrão:

Nome: todo padrão deve ter um nome significativo. Pode ser uma única palavra ou frase curta que se refira ao padrão e ao conhecimento ou estrutura descritos por ele.

Problema: estabelece o problema a ser resolvido pelo padrão, descreve a intenção e objetivos do padrão perante o contexto e forças específicas.

Contexto: estabelece pré-condições dentro das quais o problema e sua solução costumam ocorrer e para as quais a solução é desejável, o que reflete a aplicabilidade do padrão.

Força: descreve os impactos, influências, restrições relevantes ao problema e como eles interagem ou são conflitantes entre si bem como quando aplicar o padrão no contexto.

Solução: são instruções que descrevem como o problema é resolvido, podendo para isso utilizar texto, diagramas e figuras.

Usos Conhecidos: são exemplos bem sucedidos do uso dos padrões em sistemas reais.

Em particular abordaremos os padrões de projeto para estabelecer soluções para a modelagem de usuário e adaptação de sistemas.

3 A Modelagem de Usuários

O modelo de usuário é uma fonte de conhecimento que contém informações, explícitas ou implicitamente adquiridas, de todos os aspectos relevantes ao usuário para serem utilizados no processo de adaptação de uma aplicação de software. Um componente da modelagem de usuário em um sistema interativo e adaptativo formula suposições sobre o usuário baseado na interação com ele, armazena-as em um sistema de representação apropriado, infere novas suposições com base nas iniciais, mantém a consistência de um conjunto de suposições e fornece a outros componentes do sistema essas suposições sobre o usuário.

O objetivo da análise e modelagem de usuários é identificar quem são os usuários e caracterizá-los, isto é, especificar quais funções exercem, quais capacidades possuem, quais os interesses e preferências.

A **Figura 2** ilustra as tarefas que compõem o processo de adaptação de um sistema. Primeiramente o sistema coleciona dados sobre o usuário, na execução da tarefa de modelagem de usuário esses dados são processados e representados na forma de modelo de usuário; posteriormente o sistema produz o efeito de adaptação baseado no modelo de usuário.

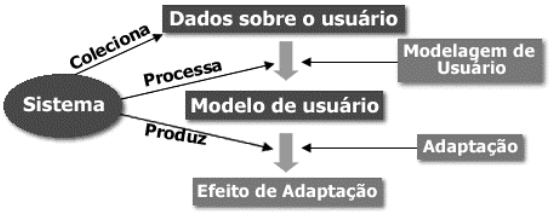


Figura 2 O processo de adaptação de um sistema

Os modelos de usuários são aplicados em sistemas, como comércio eletrônico [STRACHAN et al., 1997]; interfaces adaptativas [LANGLEY, 1999]; guias turísticos [FINK e KOBSA, 2002]; sistemas de recomendação que filtram a informação e dão sugestões aos usuários [NICK e THEMIS, 2001]; sistemas que customizam a interação de acordo com as preferências, metas, tarefas, necessidades e conhecimento do usuário [NICK e THEMIS, 2001]; sistemas tutoriais inteligentes que visam ensinar selecionando o conteúdo, o estilo e o método, de acordo com o nível de conhecimento e as características de cada aluno [BEAUMONT, 1994].

3.1 Dados que devem ser capturados dos usuários

No processo de construção de perfis de usuários, a coleta de informações sobre os mesmos é o passo inicial e de suma importância para a construção de modelos de usuários. Os principais dados que devem ser capturados dos usuários são [KOBSA, 1999]:

- Dados demográficos: nome, endereço, número do telefone; dados geográficos (código de área, cidade, estado, país); características do usuário (idade, sexo, escolaridade,estado civil); dados que indicam o estilo de vida;
- Conhecimento do usuário: são suposições sobre o conhecimento que o usuário tem sobre conceitos, relacionamentos entre conceitos, fatos e regras considerando o domínio da aplicação;
- Capacidades e habilidades do usuário: além dos sistemas tratarem com o conhecimento do usuário no domínio da aplicação (conhecimento sobre "o que"), o conhecimento do usuário sobre "o como" também exerce um importante papel para a adaptação dos sistemas às necessidades dos usuários;
- Interesses e preferências do usuário: os interesses dos usuários em uma mesma aplicação variam consideravelmente e a informação ou produto que são oferecidos a um grupo particular de usuários, pode não ser de interesse para outro grupo, como também pode até ter efeitos contrários;
- Metas e planos do usuário: os sistemas que ajudam os usuários a atingir suas metas são muito úteis, pois facilitam e aumentam a velocidade da interação, dado que o sistema tem expectativas sobre as próximas ações do usuário e pode interpretá-las de uma forma mais flexível;
- Dados de uso: os dados de uso podem ser diretamente observados e gravados ou adquiridos pela análise de dados observáveis, ou seja, pela forma como o usuário interage com o sistema.

3.2 Métodos de aquisição do modelo de usuário

Uma estratégia óbvia para adquirir informações sobre usuários é deixar que ele próprio forneça os dados necessários. A provisão de dados pelo usuário pode ser feita através de questionários feitos pelo sistema, geralmente na fase inicial do uso do sistema [KOBSA,1999].

Outro método utilizado para a geração de suposições sobre o usuário é a utilização de regras de aquisição, por exemplo regras de inferência que são executadas quando uma nova informação sobre o usuário está disponível. As regras de aquisição podem ser específicas de um dado domínio de aplicação ou podem ser independentes do domínio. Alguns sistemas usam heurísticas para definir interesses do usuário, como por exemplo: "usuário salvando A indica interesse em A" [SHARMA, 2001].

O reconhecimento de planos é outro método de aquisição que trata com o raciocínio sobre as metas que o usuário pode ter e a seqüência de ações disponíveis (plano) para que possa alcançá-las. O reconhecimento de planos consiste de tarefas que modelam possíveis ações do usuário e o relacionamento entre essas ações e do mecanismo que identifica o plano atual e as metas associadas ao usuário, obtidos através da observação de interações.

Finalmente, um método simples de aquisição para fazer uma primeira avaliação de usuários é classificá-los em categorias e então fazer predições baseadas em um estereótipo que é associado a cada categoria.

Uma vez que o modelo de usuário foi adquirido, ele precisa ser representado para fins de ser explorado posteriormente. Os modelos de usuários podem ser representados utilizando técnicas, tais como: redes bayesianas, representação baseada em lógica, parâmetros lineares, lógica fuzzy, redes neurais, ontologias [KOBSA, 1999].

3.3 Sistemas Adaptativos

Os sistemas adaptativos têm estreita relação com as tecnologias de modelagem de usuários, bancos de dados, programação distribuída na web, métodos colaborativos e interfaces dinâmicas adaptativas. Os sistemas adaptativos tentam antecipar as expectativas dos usuários a partir de modelos representando seu perfil. O objetivo geral de tais sistemas é prover seus usuários com conteúdo atualizado, subjetivamente interessante, com informação pertinente, num tamanho e profundidade adequados ao contexto e em correspondência direta com o modelo do usuário. Este funciona como uma referência para o sistema que busca adaptar seu ambiente a ele [KOBSA, 1994].

Há diferentes tipos de adaptação dependendo do grau de controle que o usuário tem sobre a adaptação. Os sistemas onde o usuário pode iniciar, propor, selecionar e produzir a adaptação ou também deixar que o sistema execute algumas dessas funções, são chamados de sistemas adaptáveis. Os sistemas que executam todas as funções acima citadas, de forma automática, são chamados de sistemas adaptativos [KOBSA, 1999].

A adaptação implícita feita de forma automática e a adaptação explícita, feita com a intervenção direta do usuário, podem coexistir na mesma aplicação. A escolha da forma de adaptação a ser usada deve ser feita cuidadosamente, levando-se em conta as conveniências e exigências dos usuários. O controle do usuário sobre a adaptação pode ser provido a nível geral (os usuários podem habilitar e desabilitar a adaptação totalmente) ou ao nível de tipo de adaptação (usuários podem aprovar ou reprovar que alguns tipos de adaptação ocorram).

O processo de personalização em aplicações pode ser dividido em três tarefas maiores que podem ser executadas por diferentes componentes dos sistemas [PALAZZO, 2003]:

- **A aquisição:** consiste em identificar informações sobre usuários, como características, comportamento de uso do computador, assim como o uso do ambiente através do monitoramento do uso do computador ou obtendo essas informações de recursos externos; fazer essas informações acessíveis ao componente de adaptação da aplicação; construir modelos iniciais dos usuários, do uso do computador e ou do ambiente (chamados de modelos de usuários, modelos de uso e modelos de ambiente).

- **A representação:** consiste em representar os conteúdos dos modelos de usuários e modelos de uso, apropriadamente, em um sistema formal, permitindo o seu acesso e posterior processamento; formular suposições sobre usuários e/ou grupos de usuários, seus comportamentos e seu ambiente, desse modo integrar informações de várias fontes.
- **A produção:** consiste em gerar a adaptação do conteúdo, da apresentação, da modalidade e da estrutura, baseada no modelo do usuário, modelo de uso e modelo do ambiente.

4 Padrões Propostos

4.1 Camadas Multiagente para Sistemas Adaptativos/adaptáveis

Nome: Camadas para sistemas adaptativos/adaptáveis

Problema: Como estruturar uma comunidade de agentes heterogêneos para que possam construir modelos de usuários e se adaptar às necessidades de diferentes tipos de usuários ou grupos de usuários?

Contexto: Um sistema necessita satisfazer necessidades e preferências de usuários individuais ou grupais, tais como apresentar o que ele quer ver, ajudar o usuário a encontrar informação, adaptar uma interface, dar ao usuário feedback sobre seu conhecimento, prever o comportamento futuro deste usuário.

Força: A organização dos agentes em camadas, segundo a similaridade de responsabilidades a serem executadas pelos agentes de cada camada, torna o sistema mais flexível, extensível e de fácil entendimento e manutenção. A alteração de uma camada não pode afetar mais que duas camadas, dado que uma camada se comunica no mínimo com uma camada e no máximo com duas. Além disso, esta organização fornece um crescimento gradual da complexidade entre as camadas, de uma forma top-down.

Solução: A solução envolve a definição de um critério para a divisão das responsabilidades entre agentes distribuídos em camadas. Os sistemas adaptativos/adaptáveis que usam a modelagem de usuários são compostos basicamente por uma camada de interface, uma camada de modelagem e uma camada de adaptação, sendo que podem ser adicionadas novas camadas de acordo com o tipo de serviço que o sistema adaptativo/adaptável irá prover ao usuário. As camadas estão organizadas de modo a fornecer e requisitar serviços às suas camadas adjacentes, como pode ser visualizado na **Figura 3**.

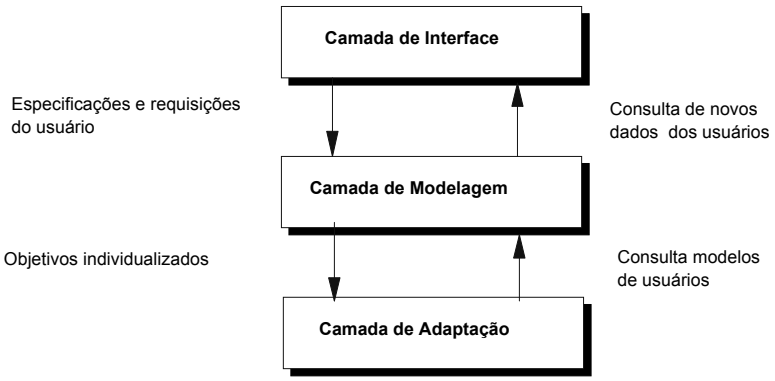


Figura 3 Padrão camadas multiagente para a modelagem de usuários

Camada de interface

A camada de interface consiste em um grupo de agentes de interface onde cada agente está associado a um usuário ou grupo de usuários com necessidades similares para auxiliá-lo(s) em muitas tarefas. Esses agentes são estruturados de acordo com o padrão interface, um tipo de padrão reativo onde as camadas Ação e Regras são especializadas para suportar as seguintes responsabilidades:

- Apresentar uma interface inicial, caso o usuário ainda não possua um modelo de usuário, ou personalizada, baseando-se no modelo de usuário corrente;
- Processar consultas;
- Colher informações sobre usuários, tais como características dos usuários, comportamento durante o uso do computador ou do sistema, através do monitoramento do uso do computador ou requisitando essas informações diretamente ao usuário;
- Dependendo da abordagem adotada, fazer perguntas ao usuário a fim de obter informações adicionais durante a realização da tarefa;
- Caso seja adotado um sistema de feedback, fazer perguntas a respeito dos resultados para possíveis melhoras na eficácia do sistema;
- Apresentar informações aos usuários de forma personalizada (respostas e explicações).

Camada de modelagem

A camada de modelagem consiste em um grupo de agentes de modelagem. Cada agente de modelagem provê serviços para um agente de interface.

Esses agentes são estruturados de acordo com o padrão modelagem, um tipo de padrão deliberativo, onde a camada Conhecimento é especializada em armazenar o modelo de usuário e as camadas Raciocínio e Ação são especializadas para suportar as seguintes responsabilidades:

- Processar a informação provida pelo usuário, através de um agente de interface e atualizar ou criar modelos de usuários, se eles ainda não existirem, de acordo com esta informação;
- Formular metas ou inferir suposições sobre usuários ou grupo de usuários com base nos modelos de usuários;
- Reformular consultas, tarefas e metas dos usuários de acordo com os modelos de usuários e enviá-las ao agente de adaptação correspondente;
- Processar resultados recebidos dos agentes de adaptação e atualizar o correspondente modelo de usuário.

Camada de adaptação

A camada de adaptação consiste em um conjunto de agentes de adaptação. Cada agente de adaptação provê serviços para o agente de modelagem. O agente de adaptação é estruturado pelo padrão adaptação, um tipo de padrão deliberativo onde a camada de Conhecimento é especializada em armazenar modelos de adaptação e as camadas Raciocínio e Ação são especializadas para suportar as seguintes responsabilidades:

- Processar as tarefas, consultas e metas dos usuários de forma personalizada de acordo com o modelo de usuário;
- Estabelecer metas para solucionar problemas de adaptação e formular um plano de ações, de acordo com os modelos de adaptação, para alcançar essas metas e para provê serviços personalizados requisitados pela camada de modelagem.

Usos Conhecidos: A organização em camadas é típica das arquiteturas multiagente para filtragem e recuperação de informação, tais como as arquiteturas RETSINA [DINIZ, 2001] [SHEHORY e SYCARA, 2000], AMALTHAEA [DINIZ, 2001] [MOUKAS e MAES, 1996] e ABARFI [DINIZ, 2001], bem como as arquiteturas de sistemas Web Adaptativos, como a

arquitetura genérica descrita por Sharma [SHARMA, 2001]. Este padrão é inspirado nas soluções arquiteturais propostas por tais sistemas.

4.2 Padrão Modelagem

Nome: Modelagem

Problema: Como criar modelos de usuários para avaliar comportamentos e conhecimentos desses usuários e manter esses modelos atualizados para que possam ser utilizados na adaptação de sistemas multiagente?

Contexto: Utiliza-se o padrão modelagem:

- Em sistemas multiagente onde há a necessidade da construção de modelos de usuários que serão utilizados como referência para fornecer serviços diferenciados às requisições e necessidades desses usuários;
- Necessita-se de um sistema com resultados diferenciados, buscando sempre respostas rápidas e pontuais;
- Necessita-se de agentes responsáveis pela tarefa de construir e manter modelos de usuários em sistemas adaptáveis/adaptativos, de forma cooperativa.

Força: Em sistemas adaptáveis/adaptativos é necessária a construção de modelos de usuários ou grupos de usuários para que esses modelos possam ser utilizados na adaptação destes sistemas.

Solução: A solução envolve a criação de um agente de modelagem do tipo deliberativo, cuja estrutura é descrita pelo padrão deliberativo, adicionando nessa estrutura conhecimento e ações específicos para que o agente seja capaz de construir e manter modelos de usuários.

Através da *camada sensitiva*, o agente de modelagem recebe requisições, especificações e dados do usuário e faz um registro do comportamento do usuário, em particular, de sua interação com o sistema.

A *camada de raciocínio* faz uma classificação dos usuários como pertencentes a um ou mais grupos e introduz as características específicas de tais grupos no modelo individual de cada usuário, de acordo com a camada de conhecimento; formula hipóteses sobre o usuário com base na história da sua interação com o sistema; generaliza a história da interação de muitos usuários em estereótipos; faz a inferência de novas hipóteses sobre os usuários, com base nos fatos iniciais; faz manutenção da consistência dos modelos de usuários e verifica novas informações comparando com informações anteriores, sempre consultando e atualizando a camada de conhecimento.

Através da *camada de ação*, o agente de modelagem executa e escalona planos elaborados pela camada de raciocínio.

Na *camada de comunicação*, o agente envia e processa mensagens.

Na *camada de conhecimento*, o agente armazena a representação de fatos sobre um ou mais tipos de características de usuários em modelos de usuários individuais (por exemplo, fatos sobre o seu conhecimento, objetivos, planos, preferências, tarefas e habilidades) e a representação das características comuns aos usuários participantes de grupos específicos em modelos grupais (estereótipos) no âmbito da aplicação.

Usos Conhecidos: Este padrão está sendo utilizado na construção de agentes de modelagem que utilizam técnicas de aquisição implícitas do modelo baseadas em algoritmos genéticos [SOBRINHO, 2003] no contexto dos projetos de pesquisa MaAE e JURIDICAS [GIRARDI, 2002].

4.3 Padrão Deliberativo

Nome: Deliberativo

Problema: Como projetar um agente para que possa raciocinar sobre um problema de forma que o possibilite atingir metas pró-ativamente dentro do contexto no qual está inserido?

Contexto: Utiliza-se o padrão deliberativo quando:

- Necessita-se que um agente esteja apto a exibir comportamento direcionado a metas tomando iniciativas e raciocinando para atingir seus objetivos;
- O agente tem que tomar decisões via raciocínio lógico;
- O agente tem que interpretar percepções, resolver problemas, fazer inferências e determinar ações;
- O agente tem que ser capaz de adaptar-se às mudanças no ambiente.

Força: Em sistemas que executem tarefas complexas, o padrão deliberativo simplifica a realização das mesmas, uma vez o agente possui conhecimento sobre o ambiente no qual está inserido e mecanismos de raciocínio para encontrar uma solução adequada.

O uso deste padrão não é indicado em aplicações que exijam respostas rápidas que não necessitem de raciocínio complexo e simbólico.

Solução: O agente deve ser do tipo deliberativo, ou seja, ele deve possuir modelos simbólicos internos de raciocínio do ambiente no qual ele está inserido e de si mesmo, ele raciocina sobre esses modelos para criar um plano para atingir suas metas. O padrão deliberativo estrutura o agente em cinco camadas organizadas verticalmente e horizontalmente: camada de comunicação, camada de ação, camada de raciocínio, camada sensitiva e camada de conhecimento. A estrutura do padrão deliberativo está representada na **Figura 4**.

Através da *camada de sensitiva*, o agente percebe do ambiente e atualiza seu conhecimento com estas percepções para refletirem o estado corrente do ambiente. Percepções são informações que o agente pode receber do mundo real, informações de recursos e interações com outros usuários. A camada sensitiva também recebe mensagens de outros agentes da sociedade.

Na *camada de raciocínio*, baseado nos seus conhecimentos, o agente elabora metas e planos para atingir essas metas.

Na *camada de ação,* o agente atua no ambiente executando ação ou ações em um plano gerado pela camada de raciocínio. A ação executada é armazenada na camada de conhecimento para refletir os efeitos das ações no ambiente. A camada de ação serve a camada de comunicação. Durante a execução de um plano, um agente pode determinar a necessidade de cooperação com outros agentes, requerendo informações ou pedindo para executar ações. Esta cooperação é suportada pela camada de comunicação.

Na *camada de comunicação*, o agente envia e processa mensagens.

A *camada de conhecimento* é um repositório onde o agente armazena informações sobre o estado corrente do ambiente através do histórico das percepções do agente e ações executadas. Esta camada contém recursos e comportamento do agente e também conhecimento sobre as habilidades dos outros agentes da sociedade.

As camadas sensitiva, raciocínio, ação e comunicação têm acesso direto a camada de conhecimento para que possam executar suas respectivas funcionalidades.

Cada camada pode ser construída através da composição de um grupo de padrões para prover a funcionalidade de cada camada.

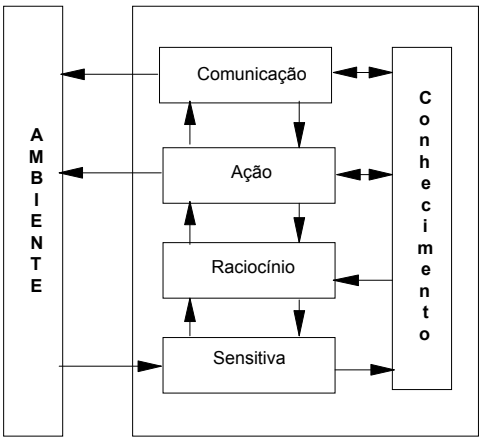


Figura 4 Estrutura do Padrão Deliberativo

O ciclo de execução do agente consiste em: na camada sensitiva, a percepção é usada para atualizar o conhecimento do agente. Na camada de raciocínio, este conhecimento é usado para determinar, dinamicamente, as metas correntes do agente para as quais um plano de ações foi construído, de acordo com as capacidades e comportamento do agente na camada de conhecimento. O plano é então executado na camada de ação. Quando um plano necessita de ações que o agente não está apto a executar porque elas não fazem parte das suas capacidades e comportamentos, a camada de comunicação determina a necessidade de cooperação com outros agentes. Então mensagens são enviadas para agentes capazes de ajudá-lo a cumprir o plano; as mensagens de outros agentes são recebidas através da camada sensitiva, depois então o ciclo completo é repetido. A camada de conhecimento é atualizada a cada novo ciclo pelas demais camadas.

Usos conhecidos: Os agentes deliberativos são amplamente utilizados em arquiteturas multiagente que utilizam raciocínio complexo [WOOLDRIGE e JENNINGS, 1994]. Kendall [KENDALL, 1997] propõe um padrão agente deliberativo que difere do padrão proposto neste trabalho principalmente em dois aspectos. Primeiro, o padrão agente deliberativo sugerido por Kendall acrescenta as camadas de mobilidade e tradução, funcionalidades das quais fazemos abstração atualmente no nosso trabalho. Segundo, o nosso padrão propõe a base de conhecimento como camada que interage verticalmente com todas as outras camadas do agente deliberativo. Infelizmente, os ambientes de desenvolvimento de sistemas multiagente, como AgentBuilder [AGENTBUILDER, 2003] e Zeus [ERICEIRA, 2001] e Jade [JADE, 2003] usam diferentes arquiteturas genéricas para a construção de agentes deliberativos. Eles provêm funcionalidades semelhantes às fornecidas pelas camadas do padrão deliberativo que nós propusemos. Porém, a forma como os módulos interagem é diferente. No agente genérico do AgentBuilder e Jade/Jess, os módulos interagem diretamente sem considerar uma hierarquia de camadas. O agente genérico do Zeus é organizado em camadas, no entanto o conhecimento do agente está distribuído em várias camadas.

5 Conclusões

A atividade de modelagem de usuários em sistemas adaptáveis é bastante complexa, mas de grande importância e aplicabilidade quando se pretende sintetizar características e habilidade de pessoas ou grupo de pessoas com o intuito de oferecer serviços personalizados.

Para lidar com a complexidade e os problemas encontrados na modelagem de usuários, utilizamos a tecnologia de agentes juntamente com os conceitos de padrões de projeto, propondo assim, padrões arquiteturais e padrões de projeto detalhado orientados a agentes. A tecnologia de agentes e os padrões foram utilizados em nosso trabalho por dois motivos principais:

- A estruturação de arquiteturas e componentes de software na forma de padrões diminui a complexidade do software em várias fases do seu ciclo de vida, pois os padrões descrevem soluções bem sucedidas para problemas recorrentes em um formato consistente e de fácil entendimento;
- O paradigma de agentes simplifica o desenvolvimento de software complexo porque os agentes podem tomar decisões sobre suas interações em tempo de execução; podem cooperar com outros agentes para atingir metas e podem acumular conhecimentos através de suas experiências.

Como trabalho futuro serão detalhadas as camadas do padrão arquitetural camadas para sistemas adaptativos/adaptáveis. Este detalhamento possibilitará a extração de novos padrões. Por exemplo, serão definidos os padrões: reativo, adaptação e interface. Para validar os padrões propostos, estes estão sendo reutilizados no desenvolvimento de frameworks multiagente que suportam a adaptação aos usuários, nos domínios jurídico e turístico. Para a construção do framework está sendo utilizada uma técnica para o desenvolvimento de sistemas multiagente baseada em ontologias e padrões que está sendo desenvolvida no contexto do projeto MaAE (Multiagent Aplication Engeneering) [GIRARDI, 2001] [GIRARDI, 2002].

6 AGRADECIMENTOS

Este trabalho é apoiado pelo CNPq.

Referências

"AgentBuilder User's guide", Reticular Systems, external documentation, disponível em:http://www.agentbuilder.com/. Acesso em 2003

BEAUMONT, I.. **User Modelling in the Interactive Anatomy Tutoring System ANATOM-TUTOR**. User Modelling and User-Adapted Interaction 4(1) 21 –45,1994.

BROOKS, R.. **A Robust Layered Control System for a Mobile Robot**. IEEE Journal of Robotics and Automation, 2 (1): 14-23, 1986.

BYLUND, Markus, WAERN, Annika. **Adaptation Agents: Proving Uniform Adaptations in Open Service Architectures,** In: Proc. of 3rd ERCIM Workshop on UI for All, 1997.

COSSENTINO, Massimo, BURRAFATO, Piermarco, LOMBARDO, Saverio, SABATUCCI, Luca. **Introducing Pattern Reuse in the Design of Multi-Agent Systems**. AITA'02 workshop at NODe02 - 8-9, 2002.

DEUGO, Dwight, WEISS, Michael, KENDALL, Elizabeth. **Reusable Patterns for Agent Coordination**. Publicado como capítulo 14 do livro: Omicini, A., Zambonelli, F., Klusch, M.,Tolksdorf, R., Coordination of Internet Agents: Models, Technologies, and Applications, Springer, 2001.

DINIZ, Alessandra. **Uma Arquitetura baseada em Agentes para a Recuperação e Filtragem da Informação**. Dissertação (Mestrado em Ciência da Computação) - Curso de Pós-Graduação em Engenharia de Eletricidade, Universidade Federal do Maranhão, 2001.

FERREIRA, Steferson, GIRARDI, Rosario. **Arquiteturas de Software baseadas em Agentes:**

do Nível Global ao Detalhado, Revista Eletrônica de Iniciação Científica da SBC, 2002.

FINK, J, KOBSA, A.. **User Modeling in Personalized City Tours.** Artificial Intelligence Review 18(1), 33-74, 2002.

GIRARDI, Rosario. **Agent-Based Application Engineering**, In: 3rd International Conference on Enterprise Information Systems (ICEIS 2001).

GIRARDI, Rosario. **Reuse in Agent-based Application Development**, In: 1st International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS´2002), International Conference on Software Engineering, Orlando, Florida, 2002.

JADE, Java Agent Development framework. Disponível em: http://jade.cselt.it. Acesso em:2003

KENDALL, Elizabeth. **The Layered Agent Pattern Language,** In: Pattern Languages of Programming (PLOP'97), 1997.

KOBSA, Alfred. **Personalised Hypermedia Presentation Techniques for Improving Online Customer Relationships**, GMD Report 66, 1999.

Kulkarni, A. **A Reactive Behavioral System for the Intelligent Room**. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 2002. Disponível em: http://citeseer.nj.nec.com/kulkarni02reactive.html. Accesso em maio 2003.

LANGLEY, P.. **User Modeling in Adaptive Interfaces**, In: J Kay (ed.) User Modeling: Proceedings of the Seventh International Conference Springer 357-370,1999.

MOUKAS, Alexandros, MAES, Pattie. **Amalthaea: Information Discovery and Filtering using a Multiagent Evolving Ecosystem**, Proceedings of the Conference on Practical Applications of Agents and Multiagent Technology, 1996.

NICK, Z, THEMIS, P.. **Web Search Using a Genetic Algorithm**. IEEE Internet Computing, 5(2), 18-26, 2001

OLIVEIRA, Ismênia. **Uma Análise de Padrões de Projeto para o desenvolvimento de Software baseado em agentes.** Monografia do Curso de Bacharel em Ciência da Computação, Universidade Federal do Maranhão, São Luís-MA, 2001.

RUSSELL, S, NORVIG, P. **Artificial Intelligence: A Modern Approach.** Prentice-Hall, 1995.

SHARMA, Amit. **A Generic Architecture for User Modeling Systems and Adaptive Web Services**, In: Workshop on E-Business & the Intelligent Web. (IJCAI, 2001), 2001.

SHEHORY, Onn, SYCARA, Katia. **The Retsina Communicator**. In Proceedings of Autonomous Agents, Poster Session, 2000.

SILVA JUNIOR, Geovanne. **Padrões Arquiteturais para o Desenvolvimento de Aplicações Multiagente**. Dissertação (Mestrado em Ciência da Computação) - Curso de Pós-Graduação em Engenharia de Eletricidade, Universidade Federal do Maranhão, 2003.

SOBRINHO, Antonio Carlos. **Uma Análise dos Algoritmos Genéticos e suas Aplicações em Sistemas de Acesso à Informação"**. Monografia do Curso de Bacharel em Ciência da Computação, Universidade Federal do Maranhão, São Luís-MA, 2003.

STRACHAN, L., ANDERSON, J., SNEESBY, M., EVANS, M.. **Pragmatic User Modelling in a Commercial Software System**, In: Proceedings of the Sixth International Conference, UM97, páginas 189-200. Springer, Vienna, New York, 1997.

WOOLDRIDGE, Michael, JENNINGS, Nicholas. **Intelligent Agents: Theory and Practice**, Knowledge Engineering Review, 1994.

Uso de Design Patterns e J2EE: um estudo de caso

Rogério Sorroche (FURB)
rs@furb.br

Maurício Capobianco Lopes (FURB)
mclopes@furb.br

Resumo. Este trabalho apresenta um estudo de caso sobre o desenvolvimento e a implementação de um software utilizando *design patterns* e a tecnologia *Java 2 Enterprise Edition*, destacando a modelagem de um sistema que agrega estas duas tecnologias.

Palavras-chave: Design Patterns, J2EE.

1 Introdução

Segundo Sun (2002) a plataforma *Java 2 Enterprise Edition* (J2EE) define um padrão para o desenvolvimento de aplicações multicamadas. Nesta arquitetura, a camada que contém as regras de negócio, normalmente implementada utilizando *Enterprise JavaBeans*, pode ficar concentrada no servidor de aplicações, sendo compartilhada com diversas aplicações clientes. As aplicações clientes não contêm a lógica do negócio, atendo-se somente à camada de apresentação. Na camada de apresentação normalmente são utilizadas as tecnologias de *Servlets* e *JavaServer Pages*.

Segundo Marinescu (2002), sem um conjunto de boas práticas de modelagem, o desenvolvimento utilizando a arquitetura multicamadas J2EE, pode se tornar muito difícil. Segundo Alur (2002), uma boa maneira de adquirir experiência em projeto é pela utilização de padrões de projeto (*design patterns*) que se constituem em um moderno e importante mecanismo para a elaboração de projetos orientados a objetos.

Sendo assim, neste trabalho é apresentada a arquitetura de um sistema de auxílio à matrícula de alunos, utilizando-se uma arquitetura multicamadas baseado no J2EE e seguindo alguns padrões de projeto específicos para esta arquitetura.

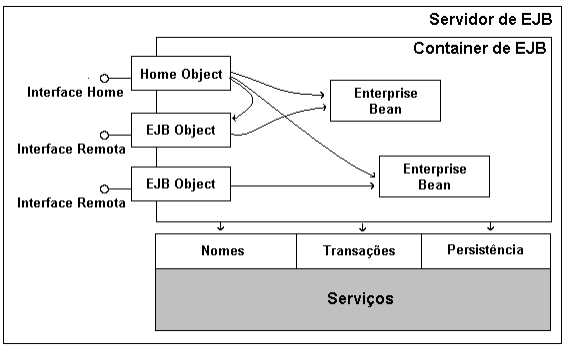
2 Java 2 Enterprise Edition (J2EE)

Segundo Sun (2002) *Java 2 Enterprise Edition* (*J2EE*) usa um modelo de aplicações multicamadas distribuído. As tecnologias que fazem parte desta plataforma de desenvolvimento são os *Enterprise JavaBeans*, que são componentes que implementam regras de negócio e os *Servlets* e os *JavaServer Pages*, que são classificados como componentes *Web*.

2.1 Enterprise Javabeans

De acordo com Roman (2002), *Enterprise JavaBeans* (*EJB*) são especificamente usados para resolver problemas de negócio. O padrão *EJB* é uma arquitetura de componentes que podem ser implantados em um ambiente distribuído. Ele especifica um contrato entre os componentes e os servidores de aplicação, de modo que um componente *EJB* pode ser implantado em qualquer servidor de aplicações que suporte *EJB*. Os *EJB* também referidos como *enterprise beans*, segundo Sun (2002), são escaláveis, transacionais e multi-usuários.

Segundo Seshadri (1999), os elementos definidos pela arquitetura de *EJB* são ilustrados na Figura 1.



Fonte: Baseado em Jubin (1999) e Roman (2002).

Figura 1 – Principais elementos da arquitetura de EJB

Os elementos desta arquitetura são:

a) servidor de *EJB*: gerencia um ou mais *containers* de *EJB* e provê os serviços de suporte, como gerenciamento de transações, persistência e acesso aos clientes. Também disponibiliza um serviço de nomes para os clientes localizarem os *enterprise beans* distribuídos através da rede (Jubin, 1999).

b) container de *EJB*: provê um ambiente em que o *EJB* possa ser executado e chamado remotamente pelos clientes.

c) *home object* e *remote home interface*: para adquirir uma referência ao objeto o cliente deve chamar uma *fábrica* de objetos. A especificação chama esta fábrica de *home object* que é uma implementação da *remote home interface* que expõe para os clientes os métodos disponíveis para criar, remover e localizar os *EJB* (Roman, 2002).

d) *EJB object* e *remote interface*: descreve os métodos de negócio, que podem ser chamados remotamente e que estão disponíveis no *bean*. Esta interface deve ser disponibilizada pelo desenvolvedor do *bean*.

2.2 Servlets

Segundo Hunter (2001) um *servlet* é uma extensão genérica de um servidor, isto é, uma classe Java que pode ser carregada dinamicamente para expandir a funcionalidade de um servidor. Geralmente os *servlets* são usados em servidores *web*, onde eles funcionam como aplicações CGI (*Common Gateway Interface*), recebendo requisições e gerando respostas.

A classe do *servlet* executa no servidor *web* por meio de um *container* de *servlets*. O servidor *web* mapeia um conjunto de URL (*Universal Resource Locator*) para um *servlet*. Quando o servidor recebe uma requisição para uma destas URL, ele repassa a requisição para o *servlet*, que gera uma resposta para o cliente. Geralmente a resposta é um documento HTML ou XML. A grande vantagem do *servlet* em relação ao *applet* é que o *servlet* executa no servidor, por isso, ele não depende da compatibilidade entre os vários navegadores de Internet (Goodwill, 2001).

2.3 Javaserver Pages

Muitas aplicações *Web* produzem primariamente páginas HTML dinâmicas que, quando acessadas, mudam somente os dados, e não a sua estrutura básica. Os *servlets* não fazem distinção entre o que é código de formatação HTML e o que são dados. A tecnologia de *JavaServer Pages* (JSP), entretanto, difere deste modelo de programação. Uma página JSP é primariamente um documento que especifica conteúdo dinâmico, ao invés de um programa que produz conteúdo. As

páginas JSP fornecem uma alternativa “centrada em documentos”, ao contrário dos *servlets* que são classes para criar conteúdo dinâmico.

Sun (2002) define uma página JSP como um documento contendo HTML estático, com algumas marcações para incluir dados ou para executar alguma lógica embutida na própria página JSP. Goodwill (2001) explica que as páginas JSP são uma extensão de *servlets*. De fato, o servidor transforma a página JSP em um *servlet* que irá gerar o conteúdo da página (Hunter, 2001).

3 Padrões de Projeto

Os padrões de projeto, também conhecidos como *Design Patterns*, visam uma melhor reutilização de software. Estes padrões tornam mais fácil reutilizar projetos e arquiteturas bem sucedidas. Eles ajudam a escolher alternativas de projeto que tornam um sistema reutilizável e a evitar alternativas que comprometam a reutilização. Segundo Marinescu (2002), um padrão de projeto é definido como uma solução desenvolvida utilizando boas práticas para um problema comum que ocorre várias vezes. Um padrão de projeto documenta e explana um problema importante que pode ocorrer no projeto ou implementação de uma aplicação e então discute a melhor solução prática para o problema.

Em termos de orientação a objetos, padrões de projeto identificam classes, instâncias, seus papéis, colaborações e a distribuição de responsabilidades. São, portanto, descrições de classes e objetos que se comunicam, implementados a fim de solucionar um problema comum em um contexto específico (Schneide, 1999). As vantagens de se utilizar padrões em um projeto são listadas por Alur (2002):

a) foram testados: refletem a experiência e conhecimento dos desenvolvedores que utilizaram estes padrões com sucesso em seu trabalho;
b) são reutilizáveis: fornecem uma solução pronta que pode ser adaptada para diferentes problemas quando necessário;
c) são expressivos: formam um vocabulário comum para expressar grandes soluções sucintamente;
d) facilitam o aprendizado: reduzem o tempo de aprendizado de uma determinada biblioteca de classes. Isto é fundamental para o aprendizado dos desenvolvedores novatos;
e) diminuem retrabalho: quanto mais cedo são usados, menor será o retrabalho em etapas mais avançadas do projeto.

Neste trabalho são apresentados padrões de projeto que descrevem a estrutura de uma aplicação e outros que descrevem os elementos do projeto. O ponto em comum entre eles é que foram desenvolvidos e aplicados especificamente à plataforma de desenvolvimento *J2EE*.

O catálogo de padrões de projeto aqui apresentados foi dividido de acordo com as camadas a que eles pertencem. A camada de apresentação contém os padrões relacionados aos *Servlets* e páginas *JSP*. Os padrões da camada de negócios são relacionados à tecnologia de *EJB* e os padrões da camada de integração estão relacionados à comunicação com recursos e sistemas externos.

Os padrões da camada de apresentação, apresentados por Alur (2002) e utilizados neste trabalho são:

a) *intercepting filter*: intercepta solicitações e respostas, e aplica algum filtro;
b) *front controller*: fornece um controlador centralizado para manter a lógica de processamento que ocorre na camada de apresentação e que, erroneamente é colocado nas visões (*view*);

c) *view helper*: encapsula a lógica que não esteja relacionada à formatação da apresentação em componentes auxiliares;

d) *composite view*: cria uma visão através da composição de outras visões;

e) *service to worker*: combina um componente distribuidor (*dispatcher*) com os padrões *Front Controller* e *View Helper*. Neste padrão a responsabilidade de recuperar o conteúdo para apresentar é do controlador;

f) *dispatcher view*: semelhante ao padrão *Service to Worker*, mas, neste padrão a recuperação do conteúdo é responsabilidade das visões.

Os padrões da camada de negócios utilizados foram (Alur, 2002).

a) *business delegate*: reduz o acoplamento entre o cliente e a camada de negócios;

b) *data transfer object*: facilita o intercâmbio de dados entre as camadas;

c) *data transfer object factory*: facilita e centraliza a criação de *data transfer objects*;

d) *session facade*: fornece uma interface unificada para um sistema distribuído;

e) *composite entity*: representa uma prática mais recomendada para criar *entity beans* de granulação grossa, agrupando os objetos dependentes em um único *entity bean*;

f) *value list handler*: gerencia o processamento e armazenamento em *cache* de consultas que retornam uma grande quantidade de informações;

g) *service locator*: encapsula a complexidade de localização e criação de serviços de negócio e localiza os objetos *Home* dos *EJB*.

Os padrões da camada de integração descritos por Alur (2002) são:

a) *data access object*: abstrai e encapsula o acesso aos dados;

b) *service activator*: facilita o processamento assíncrono para *Message-driven beans*.

As descrições detalhadas destes padrões podem ser encontradas em Alur (2002), (Marinescu, 2002) e Sorroche (2002).

4 Arquitetura do Sistema

Para exemplificar a arquitetura de um caso real de um sistema desenvolvido utilizando padrões de projeto e J2EE é apresentado um sistema de auxílio à matrícula desenvolvido por Sorroche (2002). O sistema em questão tem por finalidade apoiar o acadêmico em seu processo de matrícula, gerando sugestões de horários.

Para a visualização da arquitetura do trabalho desenvolvido são apresentados neste artigo apenas os diagramas de classe e o diagrama de estados. Os diagramas de seqüência podem ser visualizados em Sorroche (2002).

Para o desenvolvimento deste trabalho foram utilizadas as seguintes ferramentas:

a) *Together*: ferramenta case *UML*, que dá suporte a *EJB* e padrões de projeto;

b) servidor de aplicações *J2EE JBoss*: suporta a execução de *EJB* e *servlets*;

c) *JDeveloper*: ferramenta de desenvolvimento para Java;

d) *Oracle*: é o banco onde estão todas as tabelas do sistema acadêmico de graduação.

4.1 Diagrama de Classes

As classes definidas para este sistema estão separadas em pacotes, sendo eles: *ejb, cliente, dao, modelo, busca* e *web* (Figura 2).

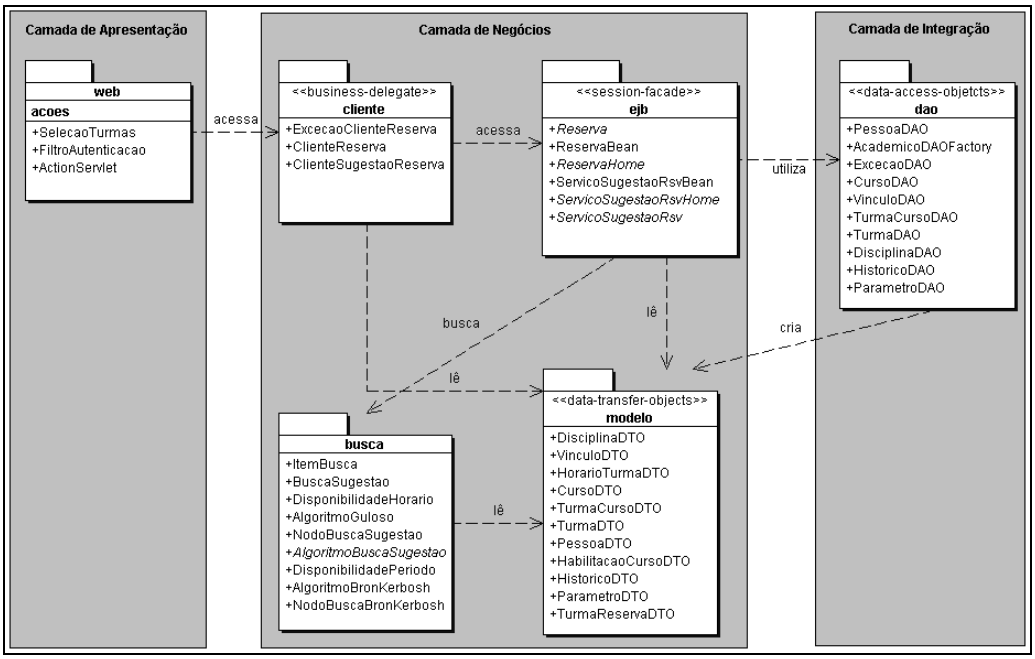


Figura 2 – Estrutura de pacotes

Cada um destes pacotes pertence a uma camada do sistema e contém os padrões de projeto para estas camadas:

a) pacote *web*: contém a implementação de um *Intercepting Filter*, que verifica a cada solicitação se o usuário efetuou a autenticação e, um *Front Controller* que age como ponto inicial de contato para efetuar o processamento da lógica na camada *web*;

b) pacote *cliente*: neste pacote estão definidos os *Business Delegate*, que são classes que representam os objetos de negócio distribuídos. Estas classes ficam do lado do cliente e simplificam o acesso aos *EJB*;

c) pacote *ejb*: contém a definição dos *EJB* que formam a camada de *Session Facade* do sistema. Os *EJB* definidos neste pacote são uma fachada para a reserva de vaga e para o serviço de elaboração de sugestões de matrícula;

d) pacote *busca*: contém a implementação dos algoritmos de busca e elaboração de sugestões;

e) pacote *dao*: contém os objetos de acesso a dados (padrão *Data Access Objects*), que extraem e encapsulam o acesso para as tabelas do sistema acadêmico;

f) pacote *modelo*: os objetos definidos neste pacote encapsulam os dados das tabelas do sistema acadêmico (padrão *Data Transfer Object*), que são passados da camada de negócios para a camada de apresentação.

A seguir serão descritos os pacotes mais detalhadamente e serão apresentados os diagramas de classes para cada pacote.

4.1.1 Pacote WEB

Neste pacote estão as classes utilizadas na camada web juntamente com as páginas *JSP* e os *servlets*. A Figura 3 apresenta o diagrama de classes deste pacote.

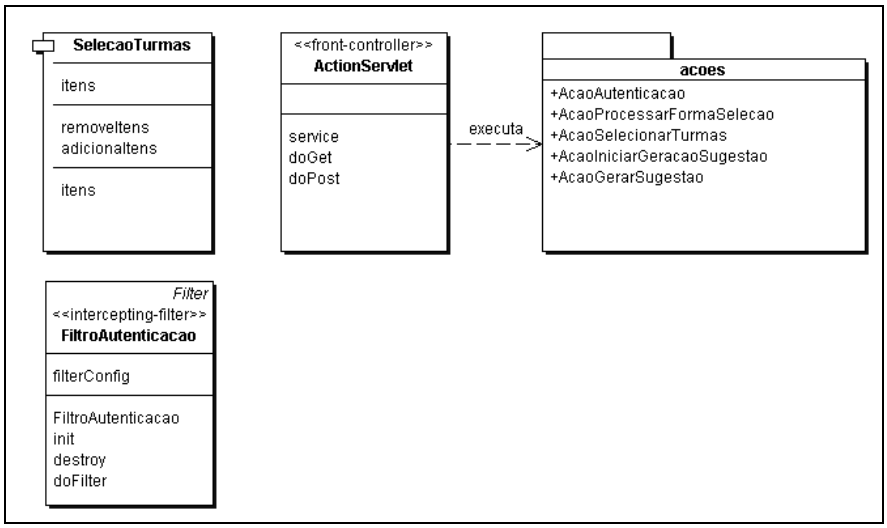


Figura 3 – Diagrama de classes do pacote *web*

A função de cada classe é descrita a seguir:

a) *SelecaoTurmas*: contém as disciplinas selecionadas para gerar sugestões;

b) *FiltroAutenticação*: é o *Intercepting Filter* para a aplicação que verifica, em todos os acessos, se o usuário efetuou a autenticação;

c) *ActionServlet*: é o *servlet* que processa a lógica de controle da camada web. Este *servlet* se adequa ao padrão *Front Controller* e processa a sua lógica através das ações (padrão *Command*). Este servlet é definido na API *Struts* (Burns, 2002).

As ações ou comandos executados pelo *ActionServlet*, estão definidos no pacote *acoes*, que é um sub-pacote de *web*. A Figura 4 apresenta o diagrama de classes para este pacote.

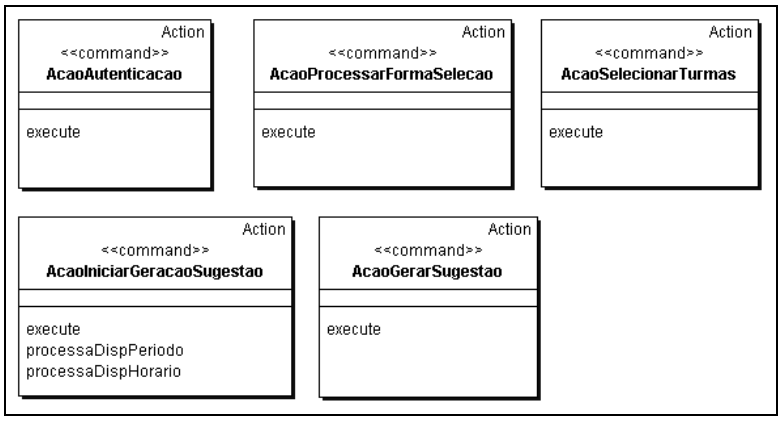


Figura 4 – Diagrama de classes para o pacote *acoes*

As classes deste pacote são descritas a seguir:

a) *AcaoAutenticacao*: é executada para autenticar o aluno no sistema validando o seu código e senha, informados na tela de autenticação;

b) *AcaoProcessarFormaSelecao*: seleciona as disciplinas que o aluno ainda não obteve aprovação e para as quais já possui os pré-requisitos;

c) *AcaoSelecionarTurmas*: seleciona as disciplinas que o aluno escolheu e armazena na sessão para uso posterior na geração das sugestões;

d) *AcaoIniciarGeracaoSugestoes*: inicia uma nova busca de sugestões com as disciplinas selecionadas;

e) *AcaoGerarSugestoes*: após iniciada uma busca, esta ação comanda a busca de sugestões para gerar as próximas sugestões.

4.1.2 Pacote Cliente

Este pacote contém os *business delegate* que abstraem o acesso aos dois *EJB* definidos anteriormente (*ReservaBean* e *ServicoSugestaoRsvBean)*, para os clientes (Figura 5).

Figura 5 – Diagrama de classes do pacote *cliente*

Estas classes têm a seguinte função:

a) *ClienteReserva*: é o *business delegate* do *EJB ReservaBean*. Os clientes acessam esta classe que por sua vez delega todas as chamadas de método para o *EJB*;

b) *ClienteSugestaoReserva*: fornece acesso ao *EJB ServicoSugestaoRsvBean*;

c) *ExcecaoClienteReserva*: exceção que pode ser gerada pelos métodos dos *business delegate*.

4.1.3 Pacote EJB

Este pacote contém os *session beans* que formam a camada de *Session Facade* do sistema. Seu diagrama de classes é apresentado na Figura 6.

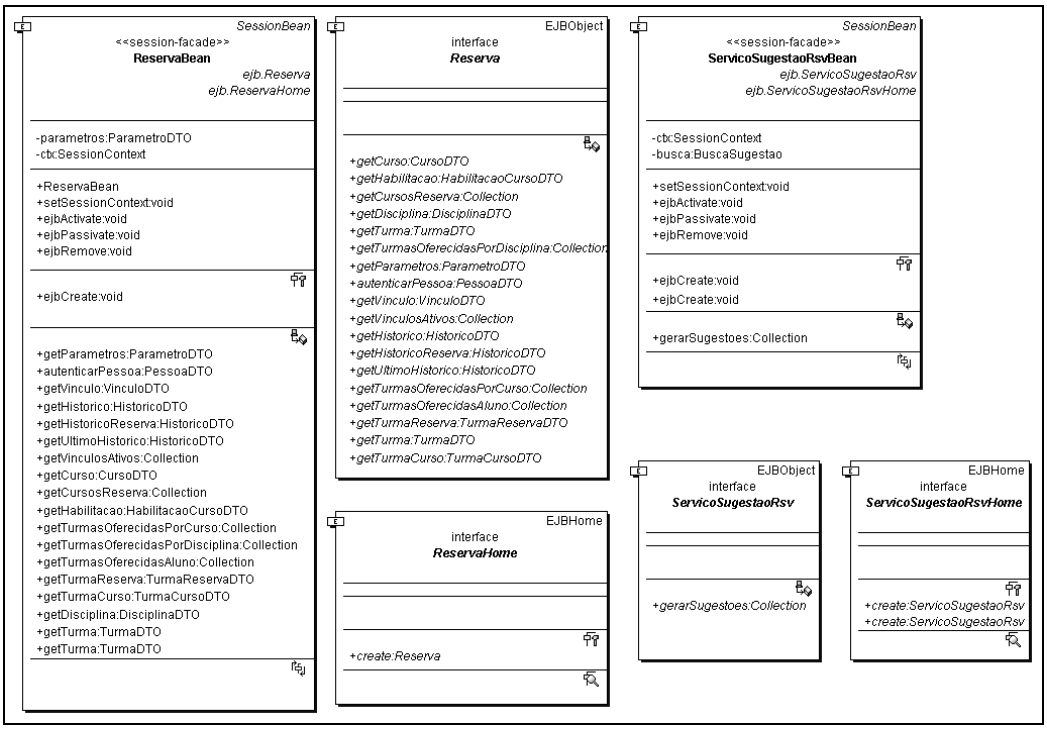


Figura 6 – Diagrama de classes do pacote *ejb*

As classes possuem as seguintes finalidades:

a) *ReservaBean*: é a classe de implementação do *EJB* que provê os serviços para a reserva de vaga. Este *EJB* contém os métodos para obter a lista de disciplinas oferecidas, autenticar um aluno e outras funções utilitárias;

b) *Reserva*: é a *remote interface* do *ReservaBean*;

c) *ReservaHome*: é a *home interface* do *ReservaBean*;

d) *ServicoSugestaoRsvBean*: é a classe de implementação do *EJB* que provê o serviço de geração das sugestões de reserva de vaga;

e) *ServicoSugestaoRsv*: é a *remote interface* do *ServicoSugestaoRsvBean*;

f) *ServicoSugestaoRsvHome*: é a *home interface* do *ServicoSugestaoRsvBean*.

4.1.4 Pacote Busca

Neste pacote está a especificação do algoritmo de elaboração das sugestões. A Figura 7 apresenta o diagrama de classes deste pacote.

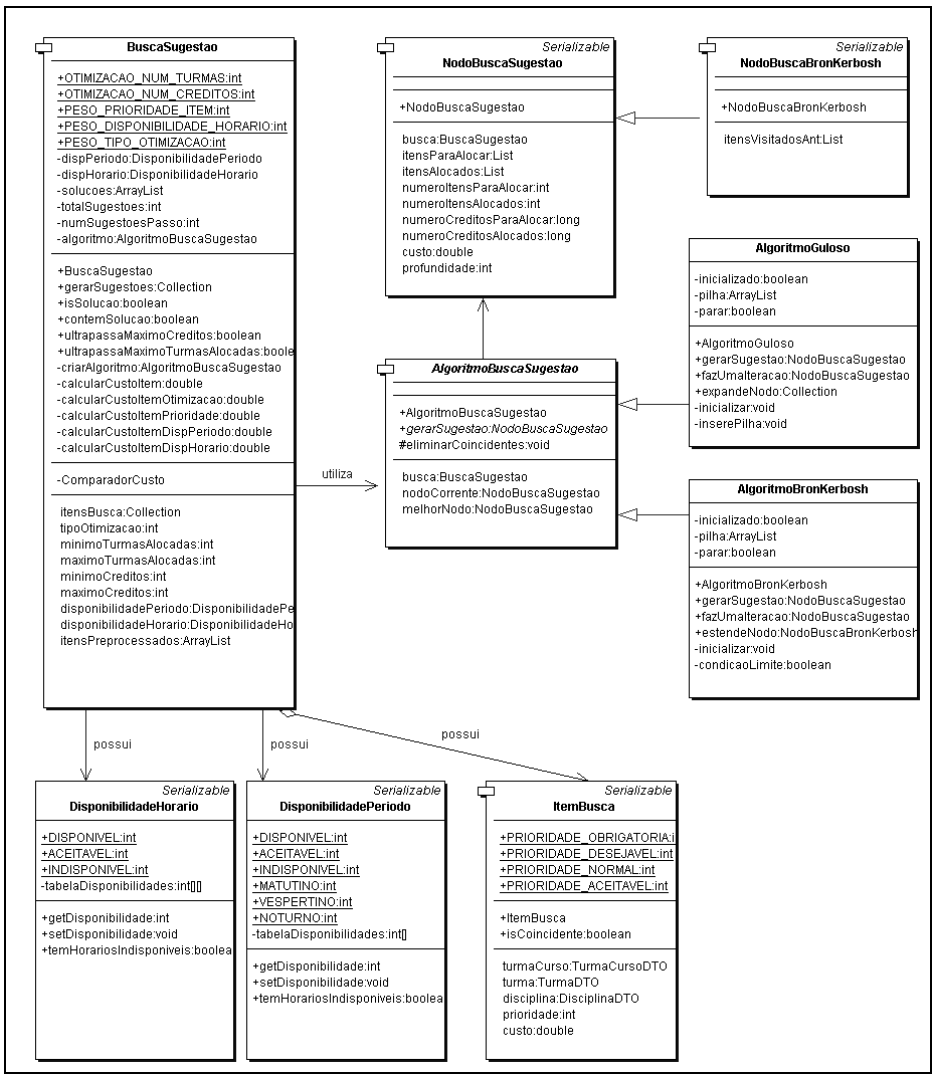


Figura 7 – Diagrama de classes do pacote *busca*

Estas classes têm as seguintes finalidades:

a) *BuscaSugestao*: representa uma busca por sugestões. Ela recebe um conjunto de objetos *ItemBusca*, as restrições, e a disponibilidade de horários do aluno e devolve um conjunto de sugestões;

b) *DisponibilidadeHorario*: representa a disponibilidade do aluno por horário;

c) *DisponibilidadePeriodo*: representa a disponibilidade do aluno por período (matutino, vespertino, noturno);

d) *ItemBusca*: representa uma disciplina ou turma com a prioridade atribuída pelo aluno e que será pesquisada na busca. Cada *ItemBusca* pode ser considerado como um vértice em um grafo que é pesquisado;

e) *AlgoritmoBuscaSugestao*: esta é uma superclasse abstrata de um algoritmo de busca de sugestões;

f) *AlgoritmoGuloso*: é a implementação do algoritmo guloso para a geração de sugestões;

g) *AlgoritmoBronKerbosh*: é a implementação do algoritmo de Bron e Kerbosh para a geração de sugestões;

h) *NodoBuscaSugestao*: representa o estado da busca a cada iteração, o algoritmo guloso utiliza esta implementação;

i) *NodoBuscaBronKerbosh*: é uma especialização de *NodoBuscaSugestao* para o algoritmo de Bron e Kerbosh, que contém um conjunto para armazenar os itens que já foram visitados.

4.1.5 Pacote Modelo

Neste pacote estão os *Data Transfer Objects*. Estes objetos servem para trocar informações entre a camada de negócios e os clientes. A Figura 8 apresenta o diagrama de classes deste pacote.

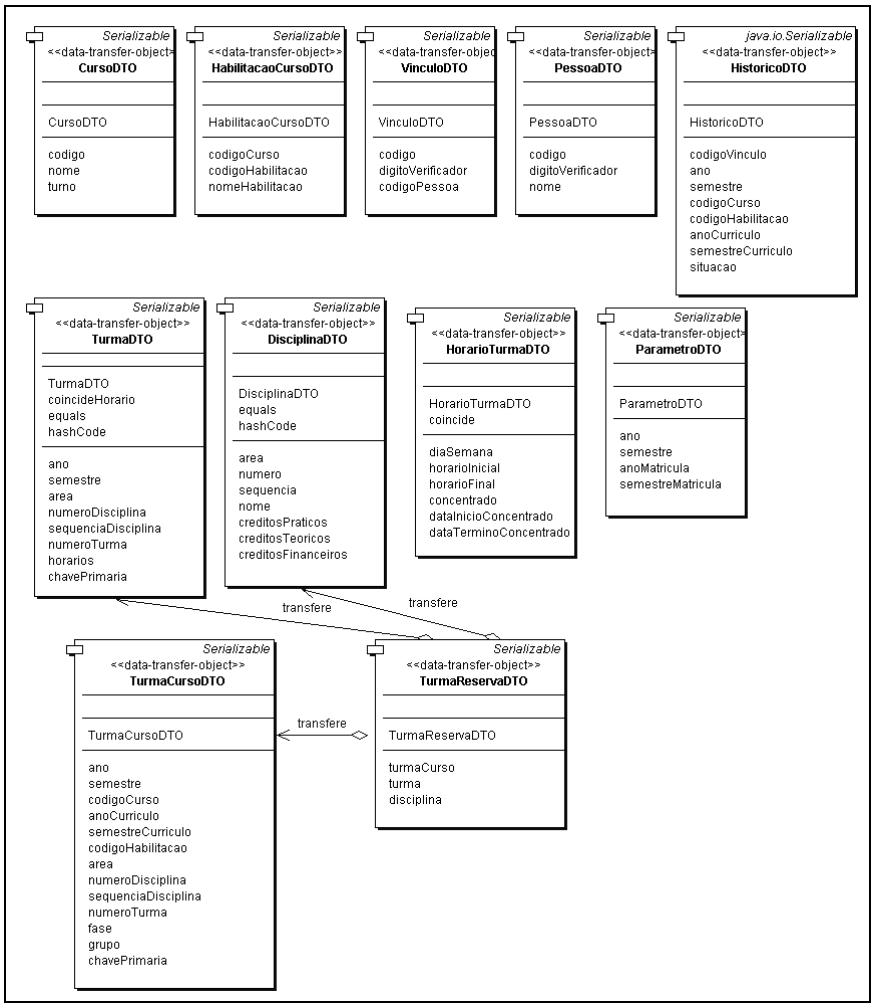


Figura 8 – Diagrama de classes do pacote *modelo*

O detalhamento da descrição destas classes pode ser obtido em Sorroche (2002).

4.1.6 Pacote DAO

Neste pacote estão definidos todos os *Data Access Objects*. Estes objetos abstraem o acesso às tabelas do banco de dados acadêmico. O diagrama de classes deste pacote é apresentado na Figura 9.

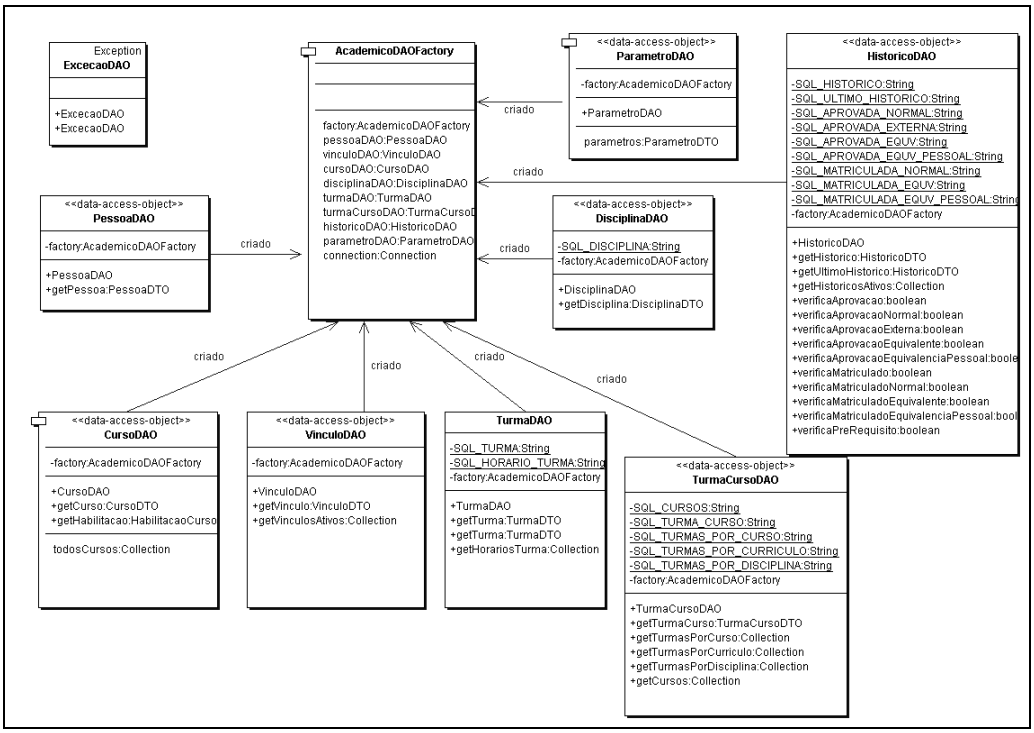


Figura 9 – Diagrama de classes do pacote *dao*

O detalhamento da descrição destas classes pode ser obtido em Sorroche (2002).

4.2 Diagrama de Estados

O diagrama de estados apresentado na Figura 10 mostra a seqüência de passos que o aluno irá seguir para gerar as sugestões. Neste diagrama as telas ou páginas JSP do sistema foram representadas como estados. Há também o estado “Gerando Sugestões”, que é o estado em que o sistema encontra-se elaborando as sugestões de reserva de vaga através do algoritmo de busca.

Neste fluxo, o aluno inicialmente se autentica no sistema e vai para uma tela onde informa as opções de seleção das disciplinas que deseja incluir na geração das sugestões. Estas opções visam incluir somente disciplinas da grade curricular do curso do aluno que ele ainda não obteve aprovação e para as quais já possua os pré-requisitos. Feita a seleção das disciplinas é apresentado um resumo para que o aluno possa definir prioridades para estas disciplinas e informar as restrições. Neste momento o aluno pode iniciar a geração das sugestões ou adicionar mais disciplinas do seu curso ou de outros cursos. O aluno pode, ainda, incluir disciplinas que façam parte da grade curricular do seu curso, mas, que estão sendo oferecidas em horários diferentes. Depois de geradas as sugestões, o aluno escolhe uma delas e efetua a sua reserva de vaga, terminando assim o processo.

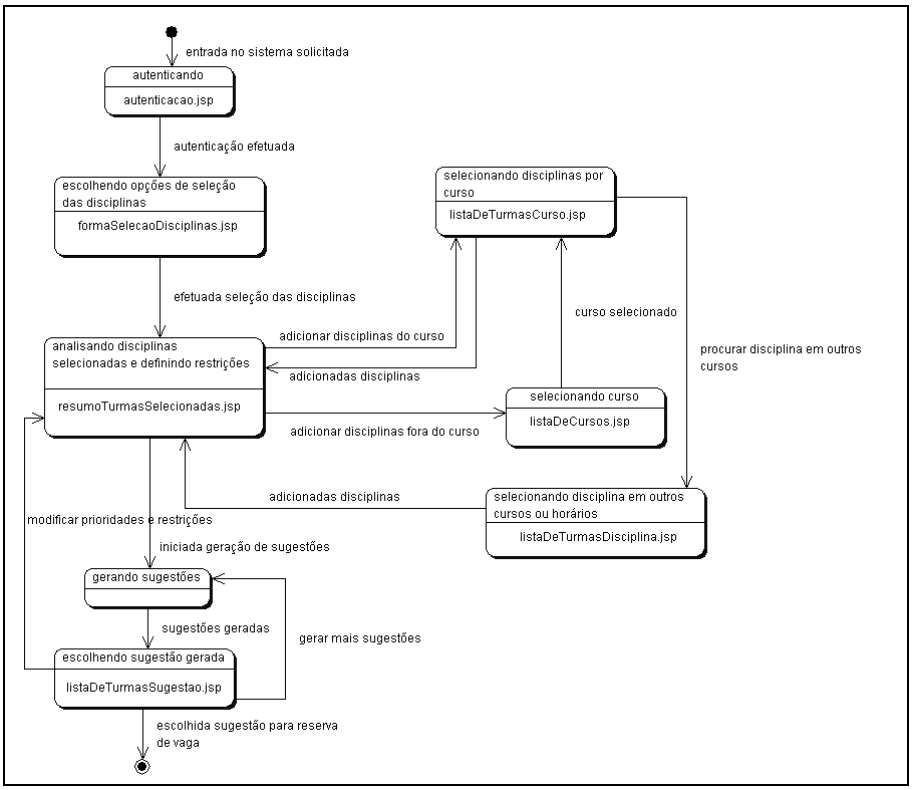


Figura 10 – Diagrama de estados

4.3 Definição da Interface do Usuário

As telas do sistema foram definidas de acordo com o padrão de projeto *Composite View*, uma *template* para as páginas *JSP*. Esta *template* é uma página *JSP* que especifica o layout de todas as páginas do sistema. A *template* define cinco seções em uma página e utiliza *custom tags* para incluir outras páginas JSP nestas seções. A Figura 11 apresenta o layout produzido pela *template*. A interface completa do sistema pode ser visualizada em Sorroche (2002).

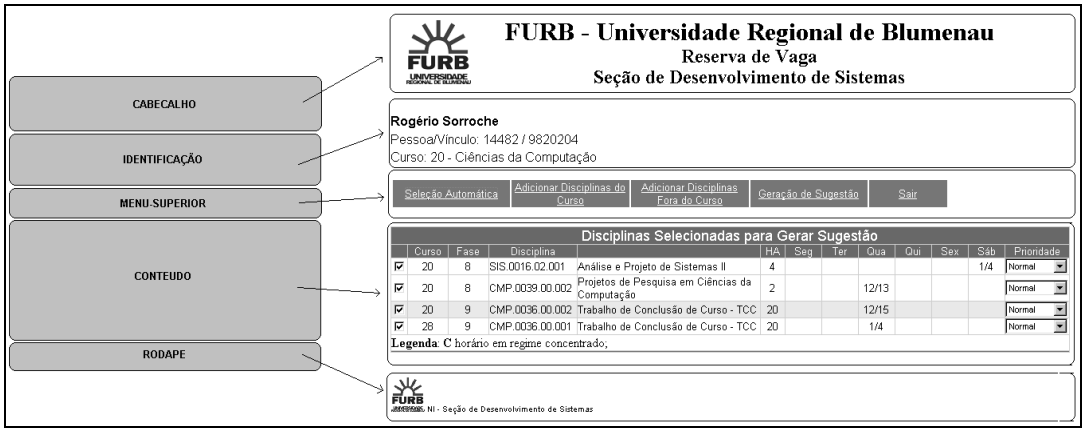

CABECALHO
IDENTIFICAÇÃO
MENU-SUPERIOR
CONTEUDO
RODAPE
FURB - Universidade Regional de Blumenau
Reserva de Vaga
Seção de Desenvolvimento de Sistemas
Rogério Sorroche
Pessoa/Vínculo: 14482 / 9820204
Curso: 20 - Ciências da Computação
Seleção Automática
Adicionar Disciplinas do Curso
Adicionar Disciplinas Fora do Curso
Geração de Sugestão
Sair
Disciplinas Selecionadas para Gerar Sugestão
Curso Fase Disciplina HA Seg Ter Qua Qui Sex Sáb Prioridade
20 8 SIS.0016.02.001 Análise e Projeto de Sistemas II 4 1/4 Normal
20 8 CMP.0039.00.002 Projetos de Pesquisa em Ciências da Computação 2 12/13 Normal
20 9 CMP.0036.00.002 Trabalho de Conclusão de Curso - TCC 20 12/15 Normal
28 9 CMP.0036.00.001 Trabalho de Conclusão de Curso - TCC 20 1/4 Normal
Legenda: C horário em regime concentrado;
FURB - NI - Seção de Desenvolvimento de Sistemas

Figura 11 – Definição de uma *template* para páginas do sistema

Utilizando esta *template*, o conteúdo das seções cabeçalho, identificação, menu-superior e rodapé, foram definidos somente uma vez para todo o sistema, como páginas *JSP*, sendo reutilizadas várias vezes.

5 Conclusões

A junção das tecnologias de padrões de projeto e J2EE em um mesmo sistema apresentou-se totalmente apropriada. Mais especificamente no sistema em questão, o uso de padrões de projeto ajudou a reduzir a complexidade e promoveu uma grande reutilização no desenvolvimento. No caso de um sistema de porte médio, conforme verificado na arquitetura apresentada neste trabalho, estas características ficam ainda mais realçadas, uma vez que na própria especificação das classes é realizada a sua relação com os padrões de projeto.

A plataforma de desenvolvimento *J2EE* apresentou uma série de vantagens para o desenvolvimento de aplicações que necessitam de escalabilidade, disponibilidade e performance. Também destaca-se a portabilidade, uma vez que o protótipo foi desenvolvido numa plataforma *Windows NT* e posto em produção numa plataforma *Linux* sem que qualquer alteração fosse feita.

Enfim, uma metodologia que agrupe padrões de projeto e J2EE pode ser um diferencial de produtividade para o desenvolvimento de soluções empresarias.

6 Referências Bibliográficas

ALUR, Deepak; CRUPI, John; MALKS, Dan. **Core j2ee patterns**: as melhores práticas e estratégias de design. Tradução Altair Dias Caldas de Moraes, Cláudio Belleza Dias, Guilherme Dias Caldas Moraes. Rio de Janeiro: Campus, 2002.

BURNS, Ed; HUSTED, Ted; MCCLANAHAN, Craig R. **Struts user guide**, [2002?]. Disponível em: < http://jakarta.apache.org/struts/userGuide/index.html>. Acesso em: 15 nov. 2002.

GOODWILL, James. **Developing java servlets**. Indianápolis: Sams Publishing, 2001.

HUNTER, Jason; CRAWFORD, William. **Java servlet programming**. Cambridge: O'Reilly, 2001.

JUBIN, Henri; FRIEDRICHS, Jürgen; TEAM, Jalapeño. **Enterprise javabeans by example**. New Jersey: Prentice Hall PTR, 1999.

MARINESCU, Floyd. **EJB design patterns**: advanced patterns, processes, and idioms. New York: John Wiley & Sons, 2002.

ROMAN, Ed; AMBLER, Scott; JEWELL, Tyler. **Mastering enterprise javabeans**. New York: John Wiley & Sons, 2002.

SCHNEIDE, Ricardo Luiz. **Design patterns**. Rio de Janeiro, maio 1999. Disponível em: < http://www.dcc.ufrj.br/~schneide/PSI_981/gp_6/design_patterns.html>. Rio de Janeiro, nov. 1999. Acesso em: 03 out 2002.

SESHADRI, Govind. **Enterprise java computing**: applications and architeture. Cambridge: Sign Books, 1999.

SORROCHE, Rogério. **Sistema de auxílio à matrícula de alunos utilizando *java 2 enterprise edition***. 2002. 108 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

SUN, Sun Microsystems. **Designing enterprise applications with the j2ee platform enterprise edition**. [S.l.], 2002b. Disponível em: <http://java.sun.com/blueprints/guidelines/ designing_enterprise_applications_2e/index.html>. Acesso em: 26 ago. 2002.

Ferramentas que auxiliam o desenvolvimento da lógica de programação.

Rafael de Santiago (UNIVALI)
santiago@inf.univali.br

Rudimar Luís Scaranto Dazzi (UNIVALI)
rudimar@inf.univali.br

Resumo. Para quem esta ingressando em uma faculdade de Ciência da Computação, a lógica é um ponto crucial. Como fator de aprendizado desta, é necessário que o aluno pratique consecutivamente exercícios que contenham algoritmos. Para melhor desenvolvimento e aplicação da lógica em algoritmos, utiliza-se a estrutura de Fluxograma, Portugol e o Teste de Mesa. Apresenta-se aqui uma discussão a cerca de algumas ferramentas desenvolvidas para auxiliar nesse processo de desenvolvimento do raciocínio lógico para programação. Propostas de melhoria e aperfeiçoamento são destacadas como foco de continuidade de tais projetos, visando a geração de uma ferramenta integrada e flexível.

Palavras-chaves: Lógica, Algoritmos, Programação, Teste de Mesa, Fluxograma.

1 Introdução

Segundo Souza at. al (2000) a idéia de utilizar algoritmos para se controlar o computador, deve-se à Ada Augusta. Foi ela que introduziu os conceitos sobre estruturas de programação como sub-rotina, laços e salto condicional (se..então). Desde as primeiras instruções elaboradas pela Condessa Ada até os dias atuais, as linguagens de programação continuam utilizando os mesmos conceitos.

A lógica de programação é a viga mestra de um indivíduo que domina as qualidades de um bom programador. Entretanto para que a lógica de programação seja assimilada, é necessário prática contínua da mesma, principalmente quando o indivíduo é leigo nesta área.

A lógica é a arte de pensar corretamente e, visto que a forma mais complexa do pensamento é o raciocínio, a lógica estuda ou tem em vista a correção do raciocínio (FORBELONE, 1993).

A maneira como a lógica é estudada implica na formação acadêmica de um aluno do curso de Ciência da Computação. Por esse motivo as disciplinas que se propõem ensinar a lógica de programação devem dispor de ferramentas confiáveis e que proporcionam a aprendizagem prática.

Uma abordagem voltada ao ensino de lógica de programação, tende, geralmente a utilização de algoritmos. Um algoritmo consiste em um procedimento, composto por uma série de passos utilizados para resolver problemas computacionais específicos, que a partir do processamento com dados de entradas irá gerar dados de saídas (CORMEN *et al*, 1999).

Para efetuar funcionalidade em um algoritmo e verificar a integridade deste é necessário testar o algoritmo verificando o conteúdo das variáveis passo a passo. Para efetuar esta tarefa costuma-se utilizar o Teste de Mesa. Também chamado Teste Exaustivo, executa para cada instrução, uma verificação, e a amostragem do conteúdo das variáveis utilizadas no algoritmo, permitindo que o programador visualize o comportamento de todo o processo. Isso permite não apenas a comprovação do correto funcionamento, mas também detectar e corrigir com facilidade eventuais erros.

Uma proposta que vem demonstrando bons resultados no curso de Ciência da Computação da Universidade do Vale do Itajaí, parte de um resgate de um antigo método de resolver problemas

computacionais, baseados na solução da lógica do problema de forma gráfica, para tal, voltou-se a ensinar programação utilizando fluxograma. Os fluxogramas quando usados para descrever a lógica de solução, sem levar em consideração detalhes de linguagem de programação ou de interface, costuma gerar bons resultados, pois os aprendizes conseguem direcionar seus esforços apenas nos passos que levarão a solução do problema. Só depois disso é que vem a preocupação com os demais detalhes de refinamento para gerar o efetivo programa.

O perfil de um fluxograma é constituído por um conjunto de estruturas de programação, cada qual, com uma representação distinta (UCCI *et al*, 1991). Com a utilização de figuras geométricas, os fluxogramas representam estruturas lógicas de seqüência, desvio condicional e repetição condicional e o fluxo de seqüência representado por setas direcionais, o que facilita sensivelmente a visualização da solução (SOUZA, 2000). Quando este modelo é utilizado de forma estruturada, não permitindo que o fluxo de seqüência saia de um lugar para outro de forma indiscriminada, evita-se os efeitos indesejáveis nas soluções tradicionais em fluxogramas.

Obtendo a idéia principal de estruturar a programação através de fluxogramas e posterior detalhamento em português estruturado, ambos utilizando as mesmas estruturas de controle, facilita o processo de aquisição do conhecimento e principalmente a consolidação deste. Ambos utilizarão uma representação correspondente:

- Estruturas de Controle: como desvio condicional, laços de repetição;
- Tipos de Dados: o tipo de valor que será inserido na variável;
- Atribuições: inserir um valor em uma variável.
- Operações aritméticas: utilizadas para cálculos entre números e variáveis;
- Operações relacionais: para estabelecerem relação entre duas comparações;
- Variáveis: para armazenar algo na memória principal;
- Vetores: conjunto de variáveis com o mesmo nome, contendo um índice para diferenciá-las;

Para obtenção de uma ferramenta capaz de satisfazer as necessidades de lógica dos alunos, estão sendo utilizados projetos de conclusão de curso de ciência da computação. Dois são os projetos que comporão esta ferramenta:

- Aplicação Web para realizar teste de Mesa em Algoritmos: desenvolvido por Clavius Leandro Medeiros. Este é um ambiente utilizado para desenvolvimento de algoritmos em português estruturado. Este ambiente permite a execução dos algoritmos confeccionados, apresentando as instruções executadas e o conteúdo das variáveis (teste de mesa) a cada passo da execução (MEDEIROS, 2001);

- Ambiente para Teste de Mesa Utilizando Fluxograma: desenvolvido por Paula Lorena Lovera Cares, é um programa produzido com a intenção de disponibilizar ferramentas para a criação de fluxogramas, propiciando também a execução dos mesmos e visualização passo a passo desta execução e de seus respectivos resultados (Teste de Mesa) (CARES, 2002).

2 Detalhamento das Ferramentas

Serão apresentadas aqui as ferramentas citadas anteriormente em maior nível de detalhe, pretendendo-se com isso esclarecer o potencial e as restrições destas. Outra ferramenta testada foi o ASA (Animação e Simulação de Algoritmos, V. 2.41). Ferramenta esta que foi desenvolvida pelo SENAC, possui conjugado com o módulo de teste de algoritmos, um tutorial sobre lógica de

programação e algoritmos. Neste, há limitações quanto as principais estruturas de fluxo, além de utilizar uma sintaxe diferenciada a proposta em aula pelos professores do curso de ciência da computação da Univali - Itajaí.

2.1 AWTM

Esta ferramenta se refere à Aplicação Web para realizar teste de Mesa em Algoritmos (MEDEIROS, 2001). Construída através da ferramenta Delphi 5, o AWTM (Aplicação Web para realizar teste de Mesa em Algoritmos) utiliza para interpretar os algoritmos, um componente chamado de atPascal Script. Este componente pertence a empresa privada e para esta ferramenta foi utilizado uma versão *shareware,* o que gera alguns desconfortos para os usuários. Este componente não foi definitivamente registrado, por não possuir todos os requisitos desejáveis para o projeto, uma vez que possui uma serie de limitações, como, por exemplo, a utilização vetores e matrizes, estruturas não suportadas pela ferramenta.

Este projeto propunha uma aplicação que permitisse a inserção de um algoritmo em português estruturado e a execução deste algoritmo, com a apresentação dos seus resultados. Antes da execução do algoritmo é realizada a compilação deste (mais precisamente a analise léxica e sintática), permitindo a correção deste tipo de erro antes do inicio da sua execução. Isso evita que este tipo de erro ocorra durante a execução, o que pode confundir o usuário, que geralmente é bastante inexperiente no assunto. Durante a execução apenas erros de lógica poderão ocorrer e levar a resultados errados.

Conforme apresentado na figura 1, o ambiente do AWTM, é simples e de fácil utilização, possibilitando, a fácil interação entre o aluno, o interpretador e gerador de teste de mesa. No ambiente o usuário digita o algoritmo e após a compilação com sucesso deste, poderá verificar a sua execução. Cada instrução executada é destacada no algoritmo grifando-se a linha em execução com fundo azul (figura 1, esquerda) e apresentado em uma tabela (consultar figura 1, direita) o conteúdo de todos as variáveis executadas até o momento.

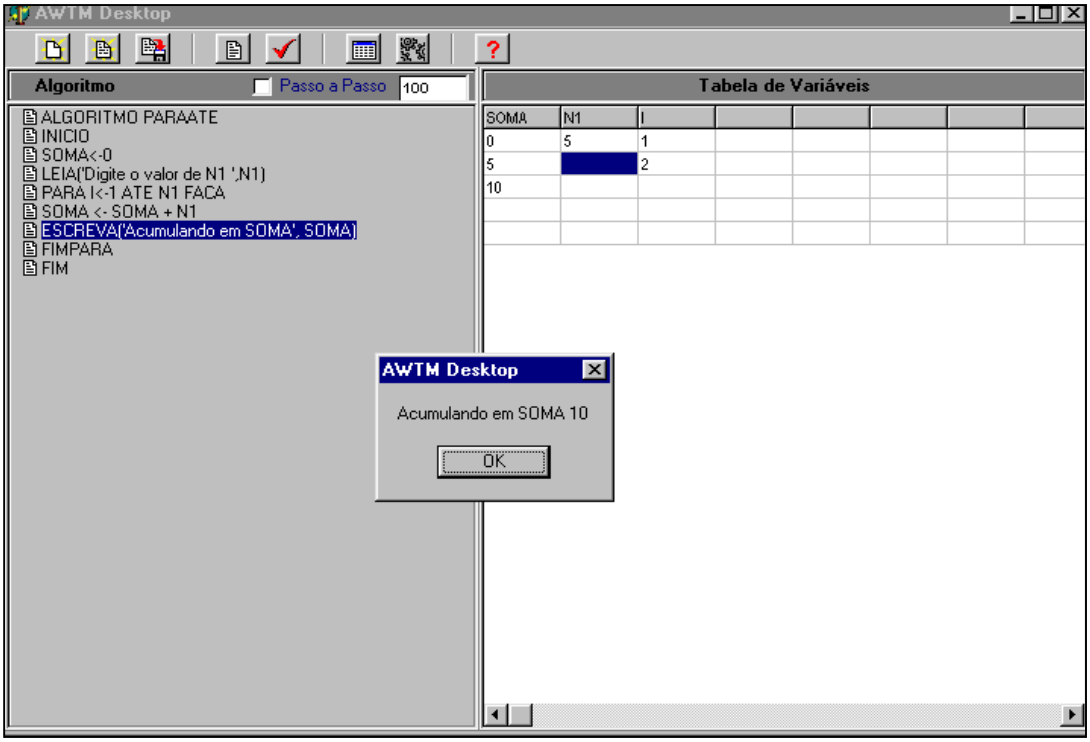


Figura 1: Ambiente do AWTM (Aplicação Web para realizar teste de Mesa).

As entradas e saídas de dados são apresentadas em uma caixa de diálogo inserida no centro da tela do ambiente (figura 1, centro) forçando o usuário a confirmar sua visualização (pressionando o botão OK) antes de prosseguir, atraindo a atenção do usuário para os resultados do seu algoritmo.

Outras configurações podem ser ajustadas, como por exemplo, o tempo de retardo entre a execução de uma instrução e outra. Esse tempo pode ser omitido e se selecionado a opção "passo a passo" irá executar uma instrução por vez e solicitar ao usuário uma confirmação para passar ao próximo passo, ou se não selecionar esta opção a execução será feita como num programa, sem paradas ou retardos.

Outro detalhe importante é a possibilidade que o aluno tem em guardar seus algoritmos em um banco de dados, podendo proteger com senha. Deste modo, um aluno pode compartilhar com outro seu conhecimento. Além disso, a ferramenta disponibiliza uma limitada, mas muito útil, ferramenta de ajuda (MEDEIROS e DAZZI, 2002).

Esta ferramenta possui algumas limitações, citadas mais acima, devido ao uso do componente atPascal Script, para interpretação e execução dos algoritmos.

2.2 ATMUF

Esta ferramenta é o Ambiente para Teste de Mesa Utilizando Fluxograma, (CARES, 2002). Para o desenvolvimento deste ambiente, foram utilizados a ferramenta Delphi 5, e o projeto AWTM (MEDEIROS, 2001). Consecutivamente a ferramenta também utiliza o atPascal Script (CARES e DAZZI, 2002).

O ATMUF (Ambiente para Teste de Mesa Utilizando Fluxograma) consiste em um ambiente gráfico utilizado para desenvolver a lógica de programação dos alunos do primeiro período, utilizando fluxogramas. Nesta ferramenta, os fluxogramas são compatíveis com as estruturas que estão presentes no portugol (também utilizado para ensino da disciplina de algoritmo do curso de ciência da computação da Univali-Itajaí).

Esta é uma ferramenta gráfica (conforme apresenta a figura 2) com a qual pretende-se desenvolver o raciocínio lógico para programação. Por ser uma ferramenta visual, facilita a compreensão do sentido do fluxo de processamento, uma vez que este é demonstrado claramente através do uso de setas de direção, que ligam as estruturas utilizadas. A simbologia utilizada para as estruturas também facilita a assimilação dos significados de suas ações no decorrer de uma solução.

Com essa ferramenta gráfica, pode-se desenvolver a lógica inicial de maneira bastante simples e visual, o que facilita a compreensão da maioria dos alunos. Com o complemento posterior, feito com os tradicionais algoritmos em portugol, faz-se o fechamento de aprendizagem desta que é uma das principais disciplinas dos cursos de computação, a disciplina de algoritmos.

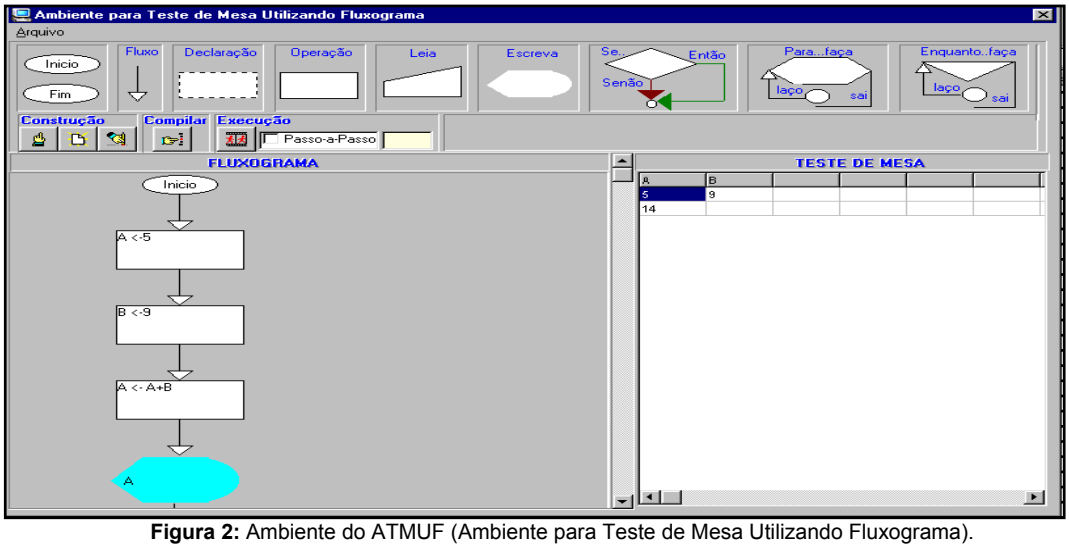


Figura 2: Ambiente do ATMUF (Ambiente para Teste de Mesa Utilizando Fluxograma).

A ferramenta permite ao usuário, um acesso rápido as estruturas, devido ao menu utilizado para construir os fluxogramas (figura 2, topo e figura 3). Basta o usuário clicar sobre a área da estrutura desejada e, em seguida, clicar sobre a área do Fluxograma (figura 2, inferior esquerdo). Para cada estrutura inserida na tela, é necessário anexar uma seta de fluxo, que demonstra para onde o fluxo do algoritmo segue. O Ambiente também permite o usuário solicitar o tempo de execução em que será testado o fluxograma, destacando a instrução que será executada na cor azul.

Figura 3: Barra de escolha da estrutura.

Assim como no AWTM, as solicitações de entrada e saídas de dados são feitas em janelas extras apresentadas no centro da tela do fluxograma. O acompanhamento da execução do fluxograma pode ser feito observando-se o conteúdo das variáveis apresentado ao lado do fluxograma (figura 2, inferior direito).

Esta pode ser uma ótima ferramenta para o aluno no desenvolvimento do aprendizado da lógica, fazendo com que entenda a maneira de como um compilador interpreta as estruturas (desvios condicionais, laços condicionais, leitura, escrita, tipos de dados).

Os pontos mais interessantes da ferramenta são a possibilidade de ter representações de estruturas, podendo assim o aluno diferencia-las facilmente, execução para cada instrução, disponibilidade de entrada e saída de dados, demonstração de teste de mesa, também atualizado a cada instrução.

Entre os pontos negativos da ferramenta estão: a impossibilidade de gravar o fluxograma, a indisponibilidade de encadear as estruturas disponíveis, a ausência de um arquivo *help* para ajudar os usuários da ferramenta.

2.3 ASA

A ferramenta ASA não será apresentada em detalhes, mas comentaremos alguns de seus aspectos. O módulo "Constructor" do ASA é aqui enfocado, pois é nele que os usuário irão construir e testar seus algoritmos.

Está ferramenta é bastante interessante, pois assim como as outras apresentadas, permite ao usuário construir e testar passo a passo suas soluções. Porém esta utiliza um formato que engloba o portugol com os fluxogramas em uma só solução, o que por um lado pode ser interessante, pois apresenta as instruções escritas (tipo um portugol) e setas direcionais interligando essas, para demonstrar a direção do fluxo de execução do algoritmo. Por outro lado, não utiliza o recurso gráfico para facilitar a assimilação do significado das estruturas de controle.

Esta possui basicamente todos os recursos apresentados nas outras duas ferramentas, porém, por ser privada não permite efetuar melhorias nem alterações e ampliações que se fazem necessárias para suprir as disciplinas que a utilizariam. As principais limitações desta, estão nas restrições dos laços de repetições (quantidade de repetições) e na falta de estruturas de dados como matrizes.

Todas as ferramentas possuem suas qualidades e defeitos, na forma como estão e foram aqui apresentadas, mas o que se pretende é o aperfeiçoamento de algumas destas (das ferramentas desenvolvidas pelo curso de ciência da computação da Universidade do Vale do Itajaí, AWTM e ATMUF), para suprir todas as necessidades das disciplinas de algoritmos do curso de ciência da computação da Univali - Itajaí, o que já esta em andamento.

3 Propostas

Pretende-se adaptar as ferramentas supracitadas para que haja um produto confiável e integro. A seguir se discutirá as mudanças necessárias.

A aplicação que se refere à construção de fluxogramas contém alguns problemas em relação à utilização das estruturas, sendo que o uso pode ficar comprometido, dependendo da situação em que foi criado o algoritmo. Outro problema referente às estruturas é a falta da possibilidade de encadeamento destas, fazendo com que o usuário tenha que mudar a solução para o problema

proposto em aula. Pretende-se então, reformular algumas estruturas e possibilitar encadeamento destas.

Referenciando, ainda o aplicativo que constrói fluxogramas, há uma carência por salvar o fluxograma feito pelo aluno. Isto deve ser implementado de maneira que a aplicação salve o fluxograma e permita que este seja aberto e utilizado novamente quando necessário. Também pretende-se aperfeiçoar o seu compilador, permitindo que a geração automática do código em portugol a partir do fluxograma e vice-versa. Num segundo passo será permitido a conversão destes fluxogramas em portugol ou em códigos de linguagem de programação, como C e Pascal.

O projeto para a finalização desta ferramenta já está em andamento, enfatizando a reestruturação da ferramenta de forma para, dentro de algum tempo, disponibilizá-la aos alunos das disciplinas que utilizará a ferramenta.

4 Conclusão

Com a intenção proposta de se prover uma ferramenta confiável ao aprendizado do aluno de primeiro período de Ciência da Computação, há ainda idéias a debater e implementar.

Pretendendo-se construir uma ferramenta com base lógica, estão sendo utilizados símbolos e estruturas que desenvolvam este raciocínio para que os aprendizes adquiram suficiente conhecimento para resolver seus problemas e fazer bom uso de todos os recursos disponibilizados pelas linguagens de programação.

Com a ferramenta de fluxogramas apresentada, pretende-se fazer a implementação de um interpretador que substitua o atPascalScript, fazendo com que as deficiências geradas por este sejam supridas.

As aplicações voltadas ao desenvolvimento de fluxogramas poderão, mais tarde, com as implementações necessárias, atender necessidades mais avançadas do curso, podendo atender até mesmo, alunos de outros períodos ou disciplinas, como, por exemplo, as disciplinas de linguagens formais e compiladores, que poderão utilizar a ferramenta como objeto de estudo e aperfeiçoamento.

Referências

CARES, Paula Lorena Lovera. **Ambiente para Teste de Mesa Utilizando Fluxograma.** Universidade do Vale do Itajaí, 2002.

CARES, Paula Lorena Lovera.; DAZZI, Rudimar Luís Scaranto. **Ambiente para Teste de Mesa Utilizando Fluxograma.** Anais do Seminário de computação FESURV/SENAC. Universidade de Rio Verde, 2002.

CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L. **Introduction to Algorithms.** Nova Yorque, 1999.

FORBELLONE, André Luiz Villar; EBERSPÄCHER; Henri Frederico. **Lógica de programação - A construção de Algoritmos e Estrutura de dados.** São Paulo : Makron Books, 1993.

GARDNER, Howard. **Inteligências múltiplas: a teoria na prática.** Porto Alegre: Artes Médicas, 1995.

MEDEIROS, Clavius Leandro. **Aplicação Web para Realizar Teste de Mesa em Algoritmo**s. Universidade do Vale do Itajaí, 2001.

MEDEIROS, Clavius Lenadro; DAZZI, Rudimar Luís Scaranto. **Aprendendo algoritmos com auxilio da WEB.** Anais do II Congresso Brasileiro de Computação. Universidade do Vale do Itajaí, 2002.

SOUZA, E. S.; GRANDI, G.; SOUZA, O. R. M.; DAZZI, R. S. D. . **Reavaliando o Ensino de Algoritmos** . Anais do Primeiro Simpósio Catarinense de Computação**.** Universidade do Vale do Itajaí, 2000.

UCCI, W.; SOUZA, R. L.; KOTANI, A. M. **Lógica de Programação: os primeiros passos**. São Paulo: Érica Ltda, 1991. p19-34.

Agentes Coletores na Gerência de Redes de Computadores

Eduardo Erlê dos Santos (UFSC)
erle@lrg.ufsc.br

Fernando Luiz Koch (Univ Utrecht)
fkoch@acm.org

Marcos Dias de Assunção (UFSC)
assuncao@lrg.ufsc.br

Carlos Becker Westphall (UFSC)
westphal@lrg.ufsc.br

Resumo. O aumento da complexidade das redes de computadores, pelo seu crescimento numérico e pela diversidade dos componentes envolvidos, tem dificultado a atividade de gerência do sistema. Uma das possibilidades para modelagem desse sistema de gerência de redes é o uso de um Grid de Agentes, que consiste em um sistema formado por comunidades de agentes distribuídos pela rede, cada um com a sua tarefa específica. Neste fluxo de informação o primeiro passo é a coleta dos dados dos dispositivos sendo gerenciados. Existe a necessidade de utilizar uma estrutura genérica de agentes de coleta que possa atuar em diferentes cenários, como no caso de atuar sobre vários tipos de protocolos de gerência, ou coletando dados de linhas de comando de vários sistemas operacionais. Este trabalho apresenta uma proposta para arquitetura e funcionamento destes agentes coletores e sua integração a um sistema de gerenciamento baseado em grids de agentes já existente.

Palavras-chave: Agentes Inteligentes, Computação Distribuída, Gerência de Redes de Computadores

1 Introdução

O aumento da complexidade das redes de computadores, pelo seu crescimento numérico e pela diversidade dos componentes envolvidos, tem dificultado a atividade de gerência do sistema. Neste ambiente, há dificuldade pela diversidade de formas de controle e monitoração visto que, embora os produtos envolvidos na rede se tornem gradativamente mais inteligentes, cada fornecedor oferece ferramentas próprias de controle para monitorar seus produtos.

Esse cenário exige um grande esforço do adminstrador da rede, levando a uma busca pela automatização do processo com o objetivo de reduzir ao máximo o esforço humano e, também, prover um método de gerenciamento mais seguro e confiável. O objetivo final é zelar pelo bom funcionamento e performance da rede de computadores e seus serviços.

Uma das possibilidades para modelagem desse sistema de gerência de redes é o uso de um Grid de Agentes, que consiste em um sistema formado por comunidades de agentes distribuídos pela rede.

Esse trabalho é motivado pela necessidade de conhecer formas de gerenciar redes de computadores do modo mais eficiente, com o menor esforço humano, baseando-se em teorias de agentes autônomos, capazes de coletar dados dos dispositivos, interpretar os dados, tomar decisões

sobre o estado da rede e aprender em novas situações, aumentando a confiabilidade do sistema e buscando prevenir problemas, em vez de pagar o alto custo de remediá-los posteriormente.

Neste fluxo de informação o primeiro passo é a coleta dos dados dos dispositivos que serão gerenciados. Apesar de contarmos com protocolos de gerenciamento bem definidos e difundidos, como por exemplo, o SNMP (Simple Network Management Protocol), existe a necessidade de construir uma estrutura genérica de agentes de coleta que possa atuar em diferentes cenários, e uma opção que supre essa necessidade é a coleta de dados através da linha de comando de vários sistemas operacionais. Os coletores devem ser configuráveis e inteligentes o suficiente para aprender novas técnicas de coleta e interpretação dos dados, conforme especificado pelos níveis mais altos do sistema de gerência.

O trabalho está dividido em cinco seções: na seção 2 é apresentada a computação em *grid, grids* de agentes e sua utilização na gerência de redes de computadores; na seção 3 é apresentada uma proposta para o funcionamento de um agente coletor nesta arquitetura; já na seção 4 são apresentados resultados práticos de uma implementação inicial e finalmente, na seção 5, são apresentadas as conclusões, bem como os trabalhos futuros a serem realizados nesta linha de pesquisa.

2 *Grids* de Agentes e Gerência de Redes de Computadores

Um *grid* pode ser visto como uma infraestrutura que possibilita o acesso e compartilhamento de recursos computacionais de uma forma fácil, segura, e em uma escala maior do que em um sistema distribuído tradicional. Um *grid* possibilita esta agregação de recursos e seu uso como se fosse uma única entidade, conforme Foster *et al* (2001).

Existem várias propostas de *middleware* e *APIs* sendo desenvolvidas para este cenário tal como o Globus, de Foster e Kesselman (1997), e o Legion, e até mesmo tecnologias baseadas em objetos distribuídos, como CORBA. Estas ferramentas podem ser usadas para proporcionar o compartilhamento e uso de recursos na escala e forma requerida em um *grid*. Os agentes são uma forma de possibilitar esta agregação e compartilhamento de recursos, criando ambientes distribuídos de alto desempenho, porém em um nível mais semântico e flexível do que as tecnologias atuais.

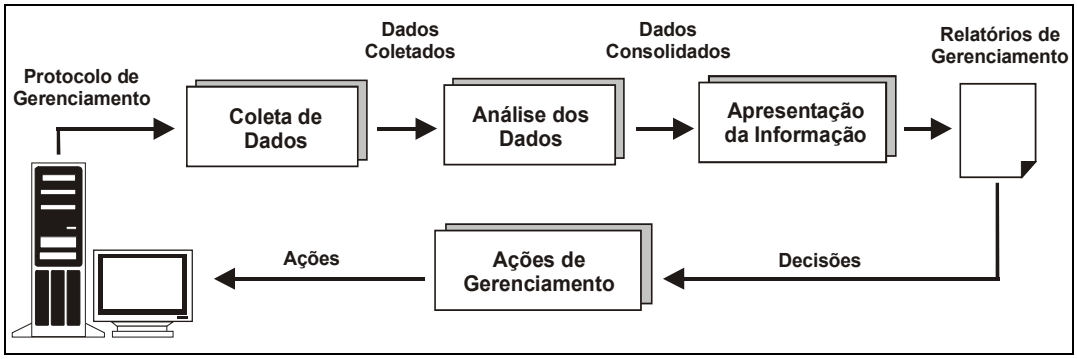


Figura 1: Fluxo do gerenciamento de redes

Conforme descrito por Koch e Westphall (2001), um sistema de gerenciamento de redes tradicional apresenta um fluxo de trabalho semelhante ao da Figura 1. Neste modelo, inicialmente, os dados são coletados dos dispositivos gerenciados da rede utilizando um protocolo de

gerenciamento ou outra interface de acesso. Posteriormente, as informações serão analisadas e, finalmente, condensadas, dando origem às verdadeiras informações de gerenciamento. Por fim, com base nestas informações, os gerentes da rede podem tomar decisões e corrigir pontos falhos no sistema.

Na gerência de redes encontramos um tipo de aplicação onde *grids* de agentes podem ser aplicados: existe um grande volume de dados que precisa ser transformado em informações de gerência, sendo que esta atividade pode requerer, além da manipulação deste grande volume de informação, um processamento intensivo. Em um *grid* podemos ter uma grande quantidade de regras de análise que conseguimos manter e utilizar. É possível efetuar uma distribuição e um balanceamento de carga da análise dos dados coletados, baseando-se na disponibilidade e capacidade dos recursos do *grid*. Caso o sistema requeira uma capacidade de processamento maior, ela pode ser adicionada ao *grid*, o que proporciona ganhos significativos e uma redução de *hardware* considerável.

2.1 Vantagens do uso de Agentes na Gerência de Redes

Conforme descrita por Koch (1997), a utilização de agentes autônomos no processo de gerência de redes de computadores introduz as seguintes características:

- *Diminuição do tráfego entre o agente e o nó gerenciável*

A redução do tráfego de rede é uma consequência natural neste modelo de gerenciamento de redes, uma vez que o processo de aquisição e análise de informações é levado mais perto do (ou mesmo no próprio) local do objeto gerenciado. O agente autônomo age como um filtro das informações coletadas do dispositivo e repassadas para os gerentes do sistema de gerenciamento.

- *Maior abstração dos objetos gerenciáveis pelos gerentes*

Tendo em vista que muitas decisões podem ser tomadas diretamente pelos agentes autônomos, algumas das características e atributos do objeto gerenciado podem ser abstraídas pelos módulos gerentes ou mesmo alguns objetos gerenciados podem ser agregados em uma unidade abstrata.

- *Maior agilidade na tomada de decisões*

Sendo que as decisões estão mais próximas dos objetos gerenciados, trazendo-se o processo de decisão para perto destes objetos evita-se a necessidade de comunicação com um sistema central.

- *Maior adaptabilidade do sistema*

O ideal dos agentes autônomos é estar preparado para quaisquer mudanças no ambiente onde estiver inserido e pronto para reagir positivamente a estas. Com agentes autônomos o nó gerenciável passa a ter "autonomia" com relação aos gerentes do ambiente, principalmente em questões não críticas. Desta forma, a gerência de rede torna-se mais automatizada.

2.2 Arquitetura do *Grid* de Agentes Para a Gerência de Redes

No trabalho de Assunção (2003), é apresentada uma arquitetura de *grid* para a gerência de redes, levando em consideração o fluxo de trabalho tradicional do gerenciamento descrito anteriormente. As atividades identificadas são: coleta de dados dos dispositivos da rede, a classificação destes dados, análise dos dados a fim de transformá-los em informações de gerência e apresentação das informações. A Figura 2 mostra um esboço da arquitetura integrando *grids* de agentes e gerência de redes de computadores.

Com base nestas atividades, foram identificados os serviços necessários e criados os *grids* que compõem o *grid* de gerência: *grid* de agentes coletores (GC), responsáveis pelas atividades de coleta de informações e por interagirem com os elementos gerenciados; *grid* de agentes classificadores e de armazenamento (GCL), que são responsáveis pelas atividades de classificação, indexação e armazenamento dos dados coletados; *grid* de agentes analisadores (GP), preocupados com a análise das informações da rede e; *grid* de interface (GI), responsável pela interação com usuário ou com outros sistemas de gerência.

3 Agentes Coletores

Dentro da aplicação apresentada anteriormente, existe o interesse de implementar um aplicativo para a primeira fase: a coleta de dados. Este aplicativo deve ser o mais flexivel possível na forma como o dado é capturado e sua interpretação inicial, encapsulando a informação em um formato padrão que é usado para transferência de dados dentro do grid. As informações extraídas podem apresentar formatos bastante heterogêneos e por isso é necessário criar uma representação comum para estes dados. Esta representação pode ser feita usando XML. Neste caso será garantido que a representação das informações coletadas da rede poderá ser interpretada corretamente pelo *grid* de agentes que irá recebê-las.

Este trabalho foi implementado utilizando-se a plataforma AgentLight, de Koch e Meyer (2003), como base de desenvolvimento para os agentes. O AgentLight foi desenvolvido em Java, rodando em ambiente Java J2SE e J2ME, utiliza uma máquina de inferência de lógica de primeira ordem – tal como PROLOG – e permite a extensão de suas funcionalidades através de criação de '*Skills*', em código Java, que podem, posteriormente, ser agregados a base de conhecimento do sistema e acessado pela máquina de inferência.

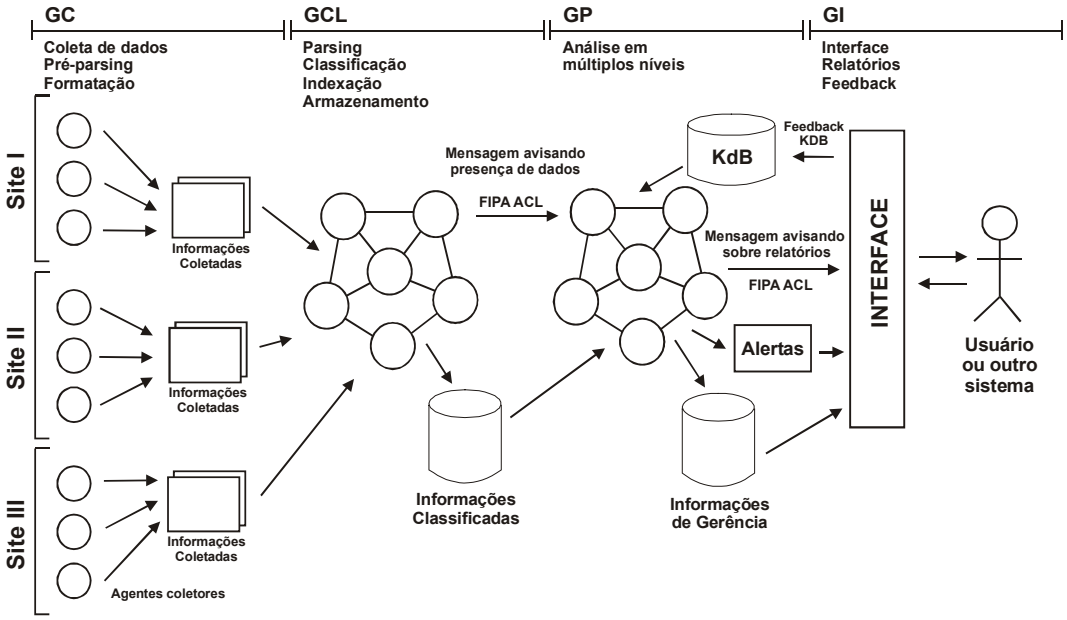


Figura 2. Arquitetura de um Grid para Aplicação de Gerência de Redes

Um agente coletor é dotado de uma base de conhecimento com regras que o permitem coletar dados usando o protocolo requerido. Estes agentes podem ter um ou mais objetivos que consistem em extrair valores de objetos gerenciados de um ou mais equipamentos da rede em intervalos de tempo. Assim como um agente pode interagir com um ou mais dispositivos da rede, pode ocorrer de vários agentes extraírem dados de um único dispositivo. O *grid* coletor pode conter agentes que executam algumas análises locais destas informações. Podemos definir agentes que tenham como objetivos localizar alguns problemas mediante a análise destes dados.

3.1 Definição dos agentes

Os agentes utilizados serão agentes reflexivos, conforme descrito por Russell e Norvig (1995). Para esse agente não existe passado nem futuro, pois as suas ações são baseadas nas informações colhidas pelo sensor naquele instante e atua no meio através das regras contidas na sua base de conhecimentos. Por isso, quando cessa a percepção do ambiente cessa a ação.

Alguns outros conceitos adotados de Koch e Westphal (2001):

- Agentes devem ser **realmente autônomos**, capazes de controlar a si mesmo e a suas ações e trabalhar independente do que aconteça com outros agentes, tendo total liberdade para adaptar-se a possíveis mudanças no ambiente.
- Agentes podem ser **guiados por metas e baseados em regras**, sendo que um conjunto de regras constitui a base de conhecimento para qualquer sistema e as metas são os objetivos de trabalho dos agentes. Definimos metas estáticas como parâmetros de configuração fixados na criação do agente, e metas dinâmicas aquelas adicionadas ao comportamento inicial do agente e que foram derivadas de mudanças do ambiente ou de mensagens recebidas.
- Agentes devem ser capazes de **aprender novas habilidades (skills**), e essas novas regras podem ser armazenadas dinamicamente nas bases locais de conhecimento dos agentes; outra possibilidade é inferir novas regras a partir do conhecimento atual do ambiente.
- Especificamente para Agentes Gerentes de Rede, **eles devem interagir com soluções comerciais usando protocolos padronizados**, ou seja, os agentes devem saber, por meio de regras em suas bases de conhecimento, como interagir com protocolos de domínio público como o SNMP (Simple Network Management Protocol) e o CMIP (Common Management Information Protocol), como resgatar informações dos arquivos de registro (log) e também como interagir com outros sistemas de gerência existentes. Tornar o seu sistema compatível significa facilidade de aceitação no mercado.

3.2 Arquitetura do agente coletor

A estrutura de um agente coletor é talvez a mais simples do conjunto de gerência da rede. O coletor é um agente reflexivo, não tem passado nem futuro, com a única função de coletar periodicamente os dados desejados de uma estação da rede, formatá-los e enviá-los à aplicação gerente que analisará os dados recebidos do coletor.

Os conhecimentos necessários para que este agente cumpra a sua tarefa seriam sobre a forma como ele vai coletar os dados, como ele vai formatá-los e como vai enviá-los. O conhecimento sobre a troca de mensagem entre agentes já está contido no agente, pois é fornecida pela plataforma sobre a qual o agente é implementado.

Para coletar os dados, primeiro é necessário definir por que meios se dará à coleta. Como exemplos podemos citar a coleta por meio de um protocolo conhecido, sendo necessário ensinar o agente a se comunicar nesse protocolo, ou por meio da interface oferecida pelo sistema

operacional do computador alvo. Os exemplos são coleta por linha de comando administrativo do sistema operacional ou protocolo de gerenciamento, como o SNMP.

Como um exemplo adotado, no método de coleta de dados por meio da linha de comando do sistema operacional não é necessário o conhecimento específico de nenhum protocolo, o que não é objetivo deste trabalho, e facilita no teste da aplicação, pois toda máquina possui um sistema operacional, mas nem toda máquina executa um programa de gerência de rede com os protocolos necessários para coleta de dados.

Para coletar os dados através de linha de comando foi utilizada uma habilidade chamada *CmdLineSkill*, que possui a regra:

```
cmdline(comando,[lista,de,parâmetros],Resultado)
```

Para formatar a resposta foi escolhida a linguagem XML, por ser uma linguagem atualmente bem aceita e de fácil utilização. Sendo assim, todos os dados colhidos serão transformados em um texto XML para serem transmitidos para os agentes encarregados de analisar os dados. Para formatar os dados em uma árvore XML, existe uma habilidade chamada 'TemplateSkill', que gera dados formatados a partir de uma estrutura fornecida como modelo. 'TemplateSkill' fornece a regra:

```
template(tree, RegExpResultados, FormatoModelo, Arvore)'.
```

Para interpretar os dados recebidos pelo sistema operacional e escrevê-los em um formato XML, utiliza-se uma habilidade chamada 'RegExpSkill' que fornece a regra `'regexp(match-add, Expressão Regular, Texto Fonte, Resultado)'`. Essa regra vai receber um 'Texto Fonte' que é a resposta obtida do sistema operacional, e de acordo com a 'Expressão Regular' apresentada como parâmetro ela vai colher as informações e retornar como uma lista.

3.3 Aprendizado

O problema da coleta de dados para um caso específico é resolvido criando uma regra de acordo com as suas características, mas esta ainda não é genérica, pois as regras só valem para um único comando do sistema operacional. Cada comando é chamado com seus parâmetros específicos e retorna uma mensagem com o seu formato, por isso, os parâmetros das regras do coletor devem se ajustar para cada situação.

Ensinar para um coletor o conjunto de regras para todos os tipos de coleta conhecidos não é um solução adequada. Por exemplo, se o ambiente da rede sendo gerenciado incluir diversos sistemas operacionais, várias regras que funcionariam em um ambiente não seriam operacionais em outros. Por outro lado, configurar manualmente determinadas regras por agente dependendo do ambiente onde este está inserido resulta em grande esforço durante o processo de inicialização e perda da flexibilidade do sistema.

Como solução a este problema o Grid fornece aos agentes uma habilidade que busca conhecimento em arquivos XML. Caso a regra para coleta de dados de um comando desejado não exista, o agente pode aprender a regra contida em um arquivo XML, obtida em um servidor conectado à Internet, possibilitando o aprendizado da regra de qualquer lugar que possua acesso a Internet.

4 Coletando dados

Durante a inicialização as regras básicas de operação são carregadas na base de conhecimento do agente coletor. Estas regras são o próprio conhecimento de operação do agente. Como descrito anteriormente, novas regras podem ser aprendidas posteriormente, mas as regras inicializadas são suficiente para as atividades de coleta de dados via SNMP, linha de comando e interpretação (*parsing*) dos dados via expressão regular. Todas estas atividades são implementadas com '*Skills*' no sistema, em Java, por razões de performance, mas acessíveis pela máquina de inferência.

As regras básicas carregadas são:

```
regexp(Cmd,Arg1,Arg2,Arg3) :-
        br.ufsc.lrg.agentgrid.skill.RegExpSkill(Cmd,Arg1,Arg2,Arg3).
    template(Cmd,Arg1,Arg2,Arg3) :-
        br.ufsc.lrg.agentgrid.skill.TemplateSkill(Cmd,Arg1,Arg2,Arg3).
    snmp(Cmd,Arg1,Arg2,Arg3) :-
        br.ufsc.lrg.agentgrid.skill.SNMPSkill(Cmd,Arg1,Arg2,Arg3).
    cmdline(Cmd,Arg1,Arg2) :-
        br.ufsc.lrg.agentgrid.skill.CmdLineExecuterSkill(Cmd,Arg1,Arg2).
    snmpcollect(Address,Community,OIDList,Result) :-
        snmp(connection,Connection,Address,Community),
        snmp(get,Connection,OIDList,Result).
```

Apresentamos aqui um exemplo de coleta via linha de comando e protocolo SNMP para demonstrar o funcionamento e a troca de mensagens no sistema.

4.1 Coleta via linha de comando

Digamos que o dado a ser coletado seja a quantidade de memória do computador, então primeiro utiliza-se uma regra para coleta por linha de comando -- neste caso, a regra 'cmdline' -- junto com um comando que retorne o dado que queremos -- por exemplo, 'systeminfo' do Windows. No final a regra composta ficará:

```
cmdline(systeminfo,['/fo','list'],Resultado).
```

E o conteúdo de *Resultado* será algo como:

```
...
Memória física total:                       255 MB
Memória física disponível:                  80 MB
Memória virtual: tamanho máximo:            875 MB
Memória virtual: disponível:                562 MB
Memória virtual: em uso:                    313 MB
...
```

A resposta é mais extensa, mas esses são os dados almejados. Então é necessário separar estes dados do resto do texto por meio de expressões regulares. Expressões regulares auxiliam na busca de uma seqüencia de caracteres específica dentro de um texto. O uso dela através de habilidades tem o intuito de filtrar informação útil das respostas recebidas do sistema operacional. Para conseguir o total da memória física a regra seria:

```
regexp(match-add, 'Memória física total: (d+\s*MB)',
                  Resultado,
                  MemoriaTotal).
```

A variável 'MemoriaTotal' conterá uma lista com os valores encontrados na expressão, que no exemplo é a lista que contêm o valor único ['255 MB']. Finalmente, utilizamos o 'TemplateSkill' para formatar o dado para saída:

```
template(tree, MemoriaTotal,
               [memoria, [total, '$1']],
               Arvore).
```

O símbolo '$1' significa que naquele local da estrutura será traduzido pelo primeiro elemento da lista contida na variável 'MemoriaTotal', no exemplo o valor '255 MB'. O resultado guardado em árvore é a estrutura:

```
<memoria>
   <total> 255 MB </total>
</memoria>
```

Esta estrutura de árvore é então adicionada como 'galho' de uma super-árvore que formará o 'conteúdo' do pacote de comunicação. O pacote será, então, enviado para o agente de armazenamento, através da estrutura de comunicação da plataforma de agentes utilizada. No grid de agentes armazenadores ela é então interpretada e armazenada apropriadamente, num processo externo ao que interessa para este trabalho.

4.2 Coleta via protocolo SNMP

No caso da coleta via protocolo SNMP é necessário apenas utilizar as regras básicas de coleta de dados via protocolo SNMP, aprendidas na inicialização do sistema. Por exemplo, para monitorar um certo equipamento da rede em intervalos de 120 segundos, ensinaríamos ao agente as seguintes regras:

```
monitor :- equipment(Address,Community),
         oidToCollect(Address,Template,OIDList),
         snmpcollect(Address,Community,OIDList,ResultList),
         templateToUse(Template,TemplateTree),
         template(tree,ResultList,TemplateTree,TreeResult),
         storeTree(TreeResult),
         tree(string,TreeResult,String,_).
```

```
storeURL('http://127.0.0.1:50000/').
equipment('127.0.0.1','public').
templateToUse('template1',[data,[id,[ip,'$1'],[time,'$2']],[sn
mp,['ifInOctets.1','$3'],['ifOutOctets.1','$4']]]).
oidToCollect('127.0.0.1','template1',['1.3.6.1.2.1.2.2.1.10.1'
,'1.3.6.1.2.1.2.2.1.16.1']).
```

O objetivo (*goal*) do funcionamento é a regra:

```
container(schedule,120,collector-snmp,monitor)
```

A saída do coletor será uma estrutura como a abaixo, onde os valores das variáveis ifInOctets.1 e ifOutOctets.1 da MIB da máquina 'localhost' são coletadas e enviadas para o armazenador:

```
<data>
  <id>
     <ip>127.0.0.1</ip>
     <time>1055193363531</time>
  </id>
  <snmp>
    <ifInOctets.1>247765</ifInOctets.1>
    <ifOutOctets.1>247765</ifOutOctets.1>
  </snmp>
</data>
```

5 Conclusão

É apresentada uma forma de se criar agentes coletores que interagem com *grids* de agentes na gerência de redes de computadores. Enquanto boa parte do código está operacional, falta ainda desenvolver novas regras de coletas e estabelecer uma 'base global de conhecimento', com várias regras para vários cenários – diferentes linhas de comandos, diferentes sistemas operacionais, diferentes dispositivos, etc. Idealmente, esta base de conhecimento estaria acessível via HTTP na Internet, permitindo assim que qualquer agente coletor pudesse aprender qualquer uma das regras já programadas interativamente (*on-demand*).

No futuro, serão trabalhados exemplos mais extensos, analisando seus efeitos sobre os *grids* armazenadores e *grids* de inferência, descritos por Assunção (2003). Enquanto aquelas atividades não são do escopo deste projeto, é sempre necessário entender e estabelecer os efeitos dos dados enviados para garantir a confiabilidade do sistema.

Grids de agentes é um assunto relativamente novo e sua utilização na gerência de redes deverá ser cada vez mais explorada. É importante manter atenção especial sobre os desenvolvimentos nesta área. Algumas das propostas de desenvolvimento do trabalho de Assunção (2003) e que também dizem respeito a este trabalho são:

- Desenvolvimento de melhores protótipos, demonstrando as vantagens da utilização de *grids* de agentes através da realização de mais medições;
- Evidenciar melhor o ponto em que a utilização de um *grid* de agentes passa a ser mais vantajosa e em que ponto deixa de ser;
- Investigar melhor a utilização de agentes móveis na coleta de dados;

Ainda existe muito a ser feito e acreditamos que a continuidade e o aperfeiçoamento deste trabalho merecem atenção especial de toda comunidade científica que atua na área, por se tratar de uma abordagem nova e promissora.

Referências

ASSUNCAO, Marcos; WESTPHAL, Carlos Becker; KOCH, Fernando Luiz. **Arquitetura de Grids de Agentes Aplicado ao Gerenciamento de Redes de Computadores e de Telecomunicação**. Rio de Janeiro, 2003. In: Simposio Brasileiro De Redes De Computadores, Proceedings, Pag. 789-804.

FOSTER, I. e KESSELMAN, C. **Globus: A Metacomputing Infrastructure Toolkit**. USA, 1997. Intl J. Supercomputer Applications, 11(2):115-128.

FOSTER, I.; KESSELMAN C.; TUECKE S**. The Anatomy of the Grid: Enabling Scalable Virtual Organizations**. USA, 2001. International Journal of High Performance Computing Applications. 15(3), Pag. 200-222.

KOCH, F. L. **Agentes Autonômos para Gerenciamento de Redes de Computadores**. Florianópolis, 1997. Trabalho de Conclusão de Curso (Mestrado em Ciências da Computação). Universidade Federal de Santa Catarina.

KOCH, F. L. e WESTPHALL, C. B.. **Decentralized Network Management using Distributed Artificial Intelligence**. USA, 2001. Journal of Network and Systems Management. Plenum Publishing Corporation. p. 291-313.

KOCH, F. L.; MEYER, J-J C. **Knowledge Based Autonomous Agents for Pervasive Computing using AgentLight**. Rio de Janeiro, 2003. In: ACM/IFIP/USENIX International Middleware Conference, MIDDLEWARE 2003 WORK-IN-PROGRESS, IEEE Distributed Systems Online.

RANA, O. F. e MOREAU, L**. Issues in Building Agent-Based Computational Grids**. Oxford, UK, 2000. Multi-Agent Systems Workshop.

RUSSEL, S. e NORVIG, P. **Artificial Intelligence, A Modern Approach**. USA, 1995. Editora Prentice Hall. Pag. 31-52.

Ferramenta de Pré e Pós-processamento para Data Mining

Deborah Ribeiro Carvalho (UTP / IPARDES)
deborah@utp.br deborah@ipardes.gov.br

Marcos Bueno (UTP / IPARDES)
bueno@onda.com.br marcos.bueno@acad.utp.br

Wilson Alves Neto (UTP / IPARDES)
w_neto@terra.com.br wilson.alves@acad.utp.br

Luiz Ricardo Lopes (UTP)

Resumo. A quantidade de dados disponíveis vem crescendo assustadoramente nos últimos anos e vários fatores contribuíram para este incrível aumento. O baixo custo na armazenagem pode ser vista como a principal causa do surgimento destas enormes bases de dados. Um outro fator é a disponibilidade de computadores de alto desempenho a um custo razoável. Como conseqüência, estes bancos de dados passam a conter verdadeiros tesouros de informação e, devido ao seu volume, ultrapassam a habilidade técnica e a capacidade humana na sua interpretação Existem várias alternativas propostas na literatura de como tratar estas bases de dados, entre elas KDD e *Data Mining*. Este artigo propõe e descreve uma ferramenta para auxiliar em várias etapas do KDD, mas especificamente, em etapas de pré e pós-processamento em relação a etapa de *Data Mining*.

Palavras-chave: *Data Mining*, Aquisição de Conhecimento, Pré-processamento, pós-processamento

1 Introdução

O conhecimento[1] é de vital importância para o mundo dos negócios e, na atualidade, as empresas reagem mais rapidamente às mudanças de mercado, onde a ação ou efeito de conhecer torna-se cada vez mais crítico ao próprio negócio.

Em razão disso, o conhecimento adquirido deve ser consistente/correto, útil e compreensível para a sua correta interpretação e uso.

Empresas que detém e/ou fornecem o conhecimento adquirido com confiabilidade, rapidez e de forma organizada, têm grandes chances de permanecerem de forma competitiva no mercado.

A quantidade de dados disponíveis vem crescendo assustadoramente nos últimos anos e vários fatores contribuíram para este incrível aumento. O baixo custo na armazenagem pode ser vista como a principal causa do surgimento destas enormes bases de dados. Um outro fator é a disponibilidade de computadores de alto desempenho a um custo razoável. Como conseqüência, estes bancos de dados passam a conter verdadeiros tesouros de informação e, devido ao seu volume, ultrapassam a habilidade técnica e a capacidade humana na sua interpretação (Carvalho, 1999).

Existe necessidade de transformar estes dados em informação para que se torne apoio nas tomadas de decisão, podendo ser usada para melhorar procedimentos, detectar tendências e características disfarçadas, e até prevenir ou reagir a um evento que ainda está por vir. Este

[1] Ao ser atribuído algum significado especial a um dado, este se transforma em uma informação ou fato. Se especialistas elaboram uma norma ou regra, a interpretação do confronto entre o fato e a regra constitui um conhecimento (Sade e Souza, 1996).

processo é um dos objetivos de estudo da área de Inteligência Artificial, onde estão sendo desenvolvidas metodologias para sua automação, através de Sistemas Computacionais ou Sistemas Inteligentes baseados no conhecimento (Carvalho, 1999).

Uma grande parte destes Sistemas são desenvolvidos utilizando as técnicas e algoritmos de *Data Mining* ou mineração de dados, que é uma área da Inteligência Artificial que trata da extração de informação válida, previamente desconhecida e de máxima abrangência, a partir de grandes bases de dados. A análise automatizada e antecipada oferecida por *Data Mining* vai muito além de consulta a um banco de dados – que é fornecida pelas ferramentas de retrospectiva típicas de sistemas de apoio a decisão, como o OLAP[2] e SQL[3] -, no sentido de permitir aos usuários explorar e inferir informação útil a partir dos dados, descobrindo relacionamentos escondidos no banco de dados.

Este artigo descreve uma ferramenta que foi implementada para auxiliar nas etapas de preparação da Base de Dados que servirão de entrada aos algoritmos de *Data Mining*, bem como, na análise do conhecimento descoberto por estes algoritmos. Desta forma, pode-se caracterizar a ferramenta proposta neste trabalho como sendo uma ferramenta de pré e pós-processamento para *Data Mining*. O artigo está composto da seguinte forma: uma introdução, a seção 2 que descreve as etapas que envolvem a descoberta de conhecimento, na qual está inserida o Data Mining, a seção 3 que descreve as funções implementadas na ferramenta e finalmente uma conclusão apresentada na seção 4.

2 Revisão da Literatura

Para que o conhecimento seja extraído de forma eficiente, os dados são submetidos a um processo chamado KDD – *Knowledge Discovery in Databases*, descoberta de conhecimento em banco de dados, processo este que possui o *Data Mining* como sua principal etapa, ou seja, o núcleo do processo, onde, devido a sua importância, muitas vezes é confundido com ele (Fayyad et. al., 1996).

Neste processo como um todo, estão envolvidas várias etapas que vão desde a seleção da(s) base(s) de dados sobre a(s) qual(is) será realizado o processamento, até a disponibilização do conhecimento descoberto para o usuário. Em alto nível de abstração pode-se dizer que essas etapas fazem parte de três grandes grupos: pré-processamento, aplicação de um algoritmo de *Data Mining* e pós-processamento (Michalski e Kaufman, 1999). Na sua grande maioria, os algoritmos de *Data Mining* produzem, como parte dos resultados, informações de natureza estatística que permitem ao usuário identificar o quão correto e confiável é o conhecimento descoberto. Porém, muitas vezes isso não é suficiente para o usuário. Mesmo que o conhecimento descoberto seja altamente correto do ponto de vista estatístico, ele pode não ser de fácil compreensão pelo usuário. Por exemplo, o conjunto de regras descobertas pode ser grande demais para ser analisado, ou conter muita redundância. Além disso, o conhecimento descoberto pode não ser surpreendente, representando algum relacionamento previamente conhecido. Poucos algoritmos de *Data Mining* produzem, como parte dos resultados, uma medida do grau de compreensibilidade e de surpresa do conhecimento descoberto. Porém, essas medidas podem ser computadas na fase de pós-processamento, como uma forma de avaliação adicional da qualidade do conhecimento descoberto, complementando (e *não* substituindo) medidas estatísticas sobre o grau de correção daquele conhecimento.

---

[2] OLAP - On Line Analitical Processing - Armazenamento multidimensional de dados, em formato de cubo, que permite o rápido agregamento de dados e detalhamento de análises.

[3] SQL – Struct Query Language - Comandos de acesso a um Banco de Dados.

Sendo assim, na etapa de pós-processamento, uma das maiores preocupações é quanto à questão de identificar, dentre os padrões descobertos na fase de *Data Mining*, aqueles que são mais surpreendentes e/ou interessantes ao usuário.

Esta questão sobre a identificação de padrões surpreendentes e/ou interessantes tem sido tratada por diversos autores na literatura. Várias propostas têm sido feitas, contemplando cobertura (Quinlan, 1993), confiança e simplicidade (Major e Mangano, 1993), entre outros.

Por outro lado na etapas que antecedem a etapa de Data Mining também podem ser grandes, por exemplo a bibliografia revela que estas fases chegam a demandar até 80% do tempo total de processamento (KDD), devido às bem conhecidas dificuldades de integração de bases de dados heterogêneas (Manilla, 1994).

Essa quantidade de tempo gasta é tida como uma ingrata surpresa para analistas e uma fonte inesgotável de frustração para os mais experientes.

2.1 Pré-Processamento

O processo inicia-se a partir do conhecimento do domínio da aplicação, assim como dos objetivos a serem atingidos. A partir daí é realizada a preparação dos dados, que envolve muitas e trabalhosas tarefas num processo KDD, pois os dados devem ser relevantes ao alcance dos objetivos, limpos, consistentes e livres de excessivas nulidades.

Seleção dos Dados

A seleção dos dados é constituída de um agrupamento organizado de uma massa de dados, alvo da prospecção.

Os dados necessários para realizar o processo, estão armazenados nas bases de dados operacionais, usadas pelos sistemas de informação das empresas e nem sempre estão de acordo com as exigências definidas pelo domínio apresentado.

Juntar estas informações em uma base de dados centralizadas nem sempre é uma tarefa fácil, já que pode envolver dados de baixo nível em tabelas relacionais ou conjunto de elementos hierárquicos em sistemas relacionais. Além de poderem ser usados em diferentes unidades da empresa, o que pode ocasionar variação na qualidade dos dados. Um exemplo disto é que alguns departamentos precisam manter bases de dados de alta qualidade contendo informações consideradas vitais as suas operações, enquanto outros, têm somente subconjuntos de dados construídos sobre estas bases (Cruz, 2000).

Algumas bases de dados são atualizadas diariamente, enquanto outras contem informações datadas de vários anos. Diferentes bases de dados usadas em diversas unidades da empresa podem usar diferentes técnicas para identificar um mesmo atributo: uma através de string e outra por números. O que deixa claro que a seleção dos dados não é uma tarefa trivial.

A qualidade do conjunto de dados está diretamente relacionada ao nível de ruído encontrado nos mesmos. O ruído pode ser proveniente de dados alterados devido aos erros de digitação ou transmissão de dados contendo informações insuficientes para o reconhecimento dos padrões de conjunto de dados desprovidos dos atributos necessários à modelagem, ou contendo atributos irrelevantes a modelagem, e de dados não atualizados. O que pode ocasionar inconsistências e dados imprecisos e/ou incompletos.

Quando se dispõe de uma pequena quantidade de dados, onde todos os exemplos são importantes, tenta-se substituir o ruído por valores consistentes ao domínio em questão ou até mesmo gerar dados manualmente. Para o caso em que a quantidade de dados é grande, tenta-se eliminar os dados que contém ruído. Em ambos os casos, fazem-se necessárias a utilização de

técnicas estatísticas para detectar os campos com ruído e, de acordo com a conveniência, substituí-los ou desconsiderá-los (Gurek, 2001).

Pré-Processamento e Limpeza

A etapa da limpeza dos dados é realizada através de um pré-processamento dos dados, visando adequá-los aos algoritmos. Isso se faz através da integração de dados heterogêneos, tratamento de ausências de dados, eliminação de dados incompletos, repetição de registros, problemas de tipagem, tratamento de ruídos, que são os dados estranhos e/ou inconsistentes.

O que pode levar a ausência de dados é a indisponibilidade ou a inexistência dos mesmos. Uma situação de indisponibilidade ocorre quando não existe divulgação do dado. Um exemplo é os dados de renda de uma pessoa física em função do sigilo (Carvalho, 1999). Já a inexistência de um dado pode ocorrer quando os dados necessários não existiam na data onde foram iniciados os processos de armazenagem. Um exemplo, são os dados da população de uma determinada cidade que foi fundada a poucos anos, pois não possui os dados dos Censos populacionais anteriores, não podendo fazer parte de estudos estatísticos regionais dos anos anteriores a sua fundação.

Como mencionado anteriormente, essa etapa pode tomar até 80% do tempo necessário para todo o processo (Manilla, 1994), devido às bem conhecidas dificuldades de integração de bases de dados heterogêneas.

Quando a base de dados é muito grande, é recomendado selecionar algumas amostras randomicamente, a fim de obter uma idéia do que pode ser esperado. Quase todas as bases de dados em grandes empresas são poluídas e quando começam a ser olhadas através da perspectiva do *Data Mining*, idéias quanto a consistência dos dados mudam (Cruz 2000).

É importante salientar que o resultado desta etapa é, em geral, um arquivo completamente distinto das bases de dados originais (Gurek, 2001).

Transformação e Codificação dos Dados

Os dados pré-processados devem ainda passar por uma transformação que os armazena adequadamente, visando facilitar o uso das técnicas de *Data Mining*, pois existem diversos tipos de algoritmos e cada um necessita de uma entrada específica, além das conversões de dados, criação de novas variáveis e categorização de variáveis contínuas.

Isto é necessário quando os processos de mineração de dados são desacoplados do sistema de banco de dados. Em algumas aplicações, ferramentas avançadas de representação de conhecimento podem descrever o conteúdo de um banco de dados por si só, usando esse mapeamento como uma meta-camada para os dados.

Em muitos países, o acesso a outras bases de dados adicionais está disponível em bases comerciais e pode prover informação de uma grande variedade de assuntos, incluindo dados demográficos, tipos de seguro que a pessoa possui, entre outros. O estudo de uma situação onde companhias trocam dados para coordenar suas operações de marketing tem sido um segmento em desenvolvimento bastante recente. Privacidade é um ponto muito importante, neste caso. Jurisprudência nesta área está se desenvolvendo rapidamente. Na maioria dos países onde não é permitida a venda de dados individuais, sem a permissão do indivíduo, é permitida a venda de informações de grupos de pessoas, mesmo não sendo uma coisa desejável do ponto de vista ético (Adriaans, 1996).

Um exemplo que pode ser aplicado para definir, de forma simples, o que é a conversão de dados, é o caso da estratificação do atributo idade, de uma determinada base de dados em quinquênios, já que o algoritmo posteriormente utilizado não aceita como entrada atributos contínuos.

2.2 Pós-Processamento

Existe um grande número de propostas na literatura para "minerar" o conhecimento descoberto, ou seja, pós-processar o conhecimento descoberto pela etapa de *Data Mining*. Em geral as propostas se enquadram em duas categorias básicas: métodos subjetivos e objetivos. No método subjetivo, é preciso que o usuário estabeleça previamente o conhecimento ou crenças, a partir do qual o sistema irá minerar o conjunto original de padrões descoberto pelo algoritmo de *Data Mining*, buscando por aqueles padrões que sejam surpreendentes ao usuário. Por outro lado, o método objetivo não necessita que um conhecimento prévio seja estabelecido. Pode-se dizer que o método objetivo é *data-driven* e o subjetivo é *user-driven* (Freitas, 1999).

Os métodos selecionados para serem implementados na ferramenta proposta neste artigo se caracterizam por serem de natureza objetiva. A razão deste critério se deve ao fato das dificuldades que são inerentes a tarefa do estabelecimento a *priori* das expectativas quanto ao conhecimento, como por exemplo, a dificuldade do usuário explicitar seu conhecimento, pela grande dependência do apoio de um usuário do domínio em questão, pelo custo inerente a tal atividade, nem todo usuário tem tempo disponível para tal, etc.

3 Descrição da Ferramenta Proposta

O primeiro passo a ser seguido no processo de aquisição de conhecimento em bases de dados (KDD) é o entendimento do problema e o conhecimento do domínio da aplicação, que só será compreendido trabalhando-se com o usuário final que, além de conhecer a aplicação, tem uma boa noção dos objetivos a serem atingidos.

Dessa forma, sem uma forte ênfase na interação entre os usuários do processo (identificados como usuários finais, especialistas do domínio e analistas do processo), é pouco provável que se consiga encontrar padrões válidos e potencialmente úteis nos dados. A busca de padrões pode ser realizada sob vários paradigmas, utilizando os mais variados métodos de aprendizado, mas é imprescindível o auxílio de pessoas diretamente ligadas aos processos.

A partir deste momento, o próximo passo é a definição das fontes de dados, bem como da estratégia de pesquisa para identificar melhor os mais importantes conjuntos de dados.

É neste ponto que se inicia o processo de Pré - Processamento, que neste estudo, será definido como a junção das três primeiras fases do processo KDD:

- Seleção;
- Pré - Processamento e limpeza;
- Transformação e codificação dos dados.

Neste processo, a seleção é uma junção organizada de dados, buscando unir em apenas uma base, todas as informações que aparentam ser necessárias para a obtenção do conhecimento desejado. É uma etapa muito importante e muitas vezes difícil de ser executada, pois pode depender de dados que se encontram em diversos departamentos da empresa, que muitas vezes possuem diferentes formas de armazenamento e recuperação, assim como uma variação muito grande na qualidade dos dados, seja com relação à digitação e transmissão de dados ou até mesmo, informações insuficientes para o reconhecimento dos padrões de conjunto de dados desprovidos dos atributos necessários a modelagem, ou contendo atributos irrelevantes a modelagem, e de dados não atualizados.

3.1 Tratamento dos Valores Ausentes

Em bases de dados é muito freqüente nem todos os atributos sejam preenchidos, por exemplo,

se o domínio a ser executado um processo KDD for o de dados sócio-econômicos dos municípios brasileiros em uma dada série histórica, com certeza vários municípios que foram criados durante a década de 90 não terão dados para as décadas anteriores. Se a aplicação for no domínio de diagnóstico médico, nem todos os pacientes realizam todos os tipos de exames, e assim sucessivamente. Ou seja, é preciso tomar atenção com a realidade de que na grande maioria dos domínios estarão presentes atributos para os quais nem todas as instâncias apresentarão valores. Se os valores ausentes forem simplesmente ignorados ou excluídos, os resultados podem ser afetados seriamente, tornando-se insignificantes ou até inválidos.

Para a correta substituição de valores ausentes, deve-se primeiramente diagnosticar se o dado ausente é relevante. Se realmente for, realiza-se a substituição do mesmo. Caso contrário, exclui-se a coluna do atributo sem relevância ou, dependendo do caso, a linha contendo o registro.

Ao serem utilizados todos os dados, existe uma melhora considerável na obtenção de resultados significativos.

Na ferramenta proposta foi implementada a opção de substituir todas as ocorrências de valores ausentes pela média para o caso de atributos contínuos e pela moda para o caso de atributos categóricos.

3.2 Estratificação de Atributos

A estratificação de dados, de um modo geral, pode ser conceituada como a divisão de dados contínuos em faixas e tem o objetivo de otimizar a performance de sistemas de classificação.

O emprego dos métodos experimentais mostra-se uma alternativa onerosa, devendo ser evitada ao máximo. Desta forma, os métodos numéricos encontram-se em uma posição de destaque junto às diversas áreas de pesquisa, sendo objeto de estudo de inúmeros pesquisadores que se concentram no aprimoramento e busca de novas técnicas numéricas que satisfaçam as crescentes exigências.

A estratificação de atributos e/ou classes contínuas desenvolve métodos de discretização para a variável objetivo (classe numérica), que permitem transformar um problema de regressão num problema de classificação. A estratificação é feita tendo em vista maximizar/minimizar os parâmetros utilizados na avaliação. Esta técnica é útil também na redução da complexidade dos dados.

Este processo permite utilizar qualquer sistema de classificação em problemas de regressão. A estratificação é feita com o objetivo de otimizar a performance do sistema de classificação que será posteriormente usado.

As formas implementadas e disponíveis na ferramenta proposta são:

- Regra de Sturges

Se for considerado uma variável contínua X representando as idade de uma amostra de trinta pessoas escolhidas ao acaso:

14,16,17,18,21,22,23,25,27,28

28,29,30,31,32,34,34,37,38,39

29,40,41,42,44,45,48,50,53,55

Como a variável X é contínua, os dados serão agrupados em classes.

Para se construir uma distribuição de freqüências, deve-se primeiramente, estabelecer o número de classes. Uma das maneiras normalmente utilizadas para a determinação deste número é realizada através da fórmula de Sturges, dada por:

$K = 1 + 3{,}32 \log n$

Onde n representa o total de observações.

Após isso, determina-se a amplitude total através da fórmula: $R = X_{max} - X_{min}$

Estabelecer o intervalo, ou amplitude de classe, dada por: $h = R/K$

Estabelecer os limites inferior e superior dos intervalos de classe, sendo que o limite inferior do primeiro intervalo deve ser menor ou igual ao menor valor da série e o limite superior da última série deve ser igual ou maior que o limite inferior + a amplitude da classe.

- Quartil

O sistema permite que atributos contínuos sejam estratificados em 4 classes, sendo que os limites superiores e inferiores respectivos a cada classes são identificados a partoor de valores dos quartis do referido atributo.

- Estratificação Manual

Nesta opção a estratificação é realizada de forma manual a partir da interação do usuário com o sistema, quando é informado o número de classes, o limite inferior e superior para cada uma delas.

3.3 Segmentação da Base de Dados

Um problema que fica bem claro na hora de realizar experimentos é a forma com que os dados estão organizados. Cada método ou algoritmo de *Data Mining* impõe um formato diferente de arquivo. Algoritmos de classificação necessitam que a base de dados seja segmentada em base de treinamento e base de teste.

A princípio parece fácil, é só juntar os dados ou dividi-los nos percentuais indicados, por exemplo, 70% para base de treinamento e os restante 30% na base de teste. Mas, na prática, existe um problema quanto ao fato de que se esta segmentação não for realizada com certo cuidado ambas as bases deixaram de representar o espaço de busca representado pela base original como um todos. Desta forma a opção disponibilizada na ferramenta proposta visa, de forma aleatória, dividir a base original de dados nos percentuais solicitado pelo usuário em tempo de processamento, sempre realizando uma verificação através dos percentuais relativos para que se evite a tendência e posterior invalidez dos resultados.

3.4 Eliminação dos Registros Repetidos

Dependendo do domínio da aplicação e muito freqüente que nas ocorram diversas repetições de registros, fato este que não colabora na execução de alguns algoritmos de *Data Mining*. Desta forma a ferramenta oferece ao usuário a possibilidade de eliminação destes registros.

3.5 Identificação de Registros Contraditórios

Existem domínios de aplicações nos quais ocorrem instâncias que contradizem umas às outras, o que pode vir a prejudicar a precisão preditiva em algoritmos de classificação por exemplo. Nestes casos é importante que o usuário tenho ciência da presença ou não de situações de contradições na Base de dados que está alimentando os experimentos. A ferramenta disponibiliza a opção de que não apenas esta identificação seja realizada, bem como o usuário opte ou não, pela retirada dos exemplos que representam a contradição.

3.6 Identificação de Regras de Exceção

A ferramenta disponibiliza uma opção que permite identificar no conjunto de padrões descoberto as regras de exceção. Este método está baseado no princípio que a contradição ao senso comum, o que pode ser bastante surpreendente (Hussain et. al, 2000). Por exemplo:

$A \rightarrow X$ regra de senso comum (alta cobertura e alta confiança)

$A, B \rightarrow \neg X$ regra de exceção (baixa cobertura, alta confiança)

$B \rightarrow \neg X$ regra de referência (baixa cobertura e/ou baixa confiança)

Fica claro a partir da estrutura anterior que o item de referência B é o que explica a causa da exceção, em relação ao senso comum $A \rightarrow X$.

Regras excepcionais podem ser bastante surpreendentes. Por exemplo, antibióticos curam doenças, mas MRSA, um tipo de *staphylococci*, é resistente a antibióticos. Embora MRSA não seja uma bactéria perigosa, pacientes francos (por outras causas) às vezes morrem tomando antibióticos, os quais curam outras doenças, mas favorecem MRSA. Esse relacionamento é bastante surpreendente, pois representa a conjunção de dois eventos raros: morte por MRSA e morte por tomada de antobiótico (Suzuki, 1997).

3.7 Ambiente de Desenvolvimento

A ferramenta foi implementada em linguagem "C" padrão e está disponível para uso dos alunos e pesquisadores da Universidade Tuiuti do Paraná (UTP), além de apoiar a implementação de projetos da área sócio-econômica, sediados no Instituto Paranaense de Desenvolvimento Econômico e Social (IPARDES), que envolvem Aquisição de Conhecimento. Dada a generalidade na concepção da ferramenta, esta pode ser utilizada no apoio às diversas tarefas de *Data Mining*, como por exemplo, Classificação e Descoberta de Regras de Associação.

4 Conclusão

Este artigo propõe e descreve uma ferramenta que auxilia nas fases de pré e pós-processamento em relação a etapa de Data Mining no contexto de KDD. Esta ferramenta foi construída a partir de um projeto de Iniciação Científica, por alunos do segundo ano do curso de Ciência da Computação.

Esta ferramenta foi modelada, implementada e encontra-se atualmente em fase de testes. Estes testes estão sendo realizados por alunos que freqüentam a disciplina de Sistemas Inteligentes. Após a validação da mesma por estes alunos, serão implementadas as alterações e adaptações sugeridas, para posterior disponibilização da mesma para a comunidade em geral.

4.1 Trabalhos Futuros

Algumas sugestões para novas implementações estão sugeridas:

- implementar novas regras de estratificação de atributos contínuos assim como métodos diferentes de tratamento de valores ausentes, como por exemplo Redes Neurais (Hruschka, 2001), etc.
- disponibilizar a opção para que não apenas as regras de exceção sejam identificadas na etapa de pós-processamento, mas também que métodos de avaliação do quão surpreendente/interessante é o conhecimento descoberto pelos algoritmos de *Data Mining* (Freitas, 1998).

Referências

ADRIAANS, Pieter. Zabtinge, Dolf – *Data Mining, Addison Wesley Longmann* – England – 1996.

CARVALHO, Deborah Ribeiro. *Data Mining Através de Indução de Regras e Algoritmos Genéticos.* Dissertação para obtenção do grau de Mestre, Pontifícia Universidade Católica do Paraná – 1999.

CRUZ, Priscila Gomes Bastos. *Data Mining Através de Regra de Associação e Arvore de Decisão*. Monografia para obtenção do grau de tecnologo em Processamento de Dados, Universidade Tuiuti do Paraná – 2000.

FAYYAD, Usama M; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic; UTHURUSAMY, Ramasamy. *Advances in Knowledge Discovery and Data Mining*. USA: American Association for Artificial Intelligence. 1996.

FREITAS.A. On objective measures of rule surprisingness. *Principles of Data Mining & Knowledge Discovery (Proc. 2nd European Symp*., PKDD'98. Nantes, France, Sep. 1998). Lecture Notes in Artificial Intelligence 1510, 1-9. Springer-Verlag. 1998.

FREITAS, A. On Rule Interestingness Measures. *Knowledge – Based Systems Journal 12* (5-6), p. 309-315. 1999.

GUREK, Eleazar Lucas. *Data Mining – Aquisicao de Conhecimento em Bancos de Dados.* Monografia para obtenção do grau de tecnologo em Processamento de Dados, Universidade Tuiuti do Paraná – 2001.

HUSSAIN, F.; LIU, H.; LU, H. Exception Rule Mining with a Relative Interestingness Measure. *PAKDD*-2000, LNAI 1805, p. 86-96. 2000.

HRUSCHKA, Eduardo R. *Algoritmos Genéticos de Equipamentos para Extração de Regras de Redes Neurais*. Tese apresentada para a obtenção do grau de doutor pela Universidade Federal do Rio de Janeiro (COPPE). 2001

MANILLA, H. Finding Interesting Rules From Large Sets of Discovered Association Rules, *3rd International Conference on Information and Knowledge Management* –1994.

MAJOR, J. A.; MANGANO, J.J. Selecting Among Rules Indiced from a Hurricane Database, *Proc. AAAI 93, Workshop Knowledge Discovery in Databases,* p. 28-41. 1993.

MICHALSKI, Ryzzard; KAUFMAN, Kenneth. *Data Mining and Knowledge Discovery: A Review of Issues and Multistrategy Approach,* In: Ryszard S. Michalski; Ivan Bratko and Miroslav Kubat (Eds.). 1998.

QUINLAN, J. R. *C4.5 Programs for Machine Learning*, Morgan Kaufmann Publisher, USA. 1993.

SADE, Alberto Sulaiman; SOUZA, Jano Moreira de. *Prospecção de Conhecimento em Bases de Dados Ambientais* - 1996.

SUZUKI, E.Autonomous discovery of reliable exception rules. *Proc. 3rd Int. Conf. Knowledge Discovery & Data Mining*, 259-262. AAAI Pres, 1997.

Computador de Bordo Automotivo baseado em PC-Linux

Cristiano Freese (FURB)
freese@inf.furb.br

Miguel Alexandre Wisintainer (FURB)
maw@furb.br

Resumo. Este trabalho irá demonstrar a integração entre tipos diferentes de *hardware* e *software*, representando funcionalidades de um computador de bordo automotivo. Funções como indicação de temperatura ambiente, velocidade veicular, status de carga da bateria, interpretação de comandos via controle remoto e interface homem/máquina são desempenhadas por um *hardware* microcontrolado da família PIC denominado interface, enquanto que funções como registro de excessos de velocidade e reprodução de músicas em formato MP3 são desempenhadas por um *hardware* PC denominado PC, utilizando um formato de placas-mãe denominado Mini-ITX. A execução de arquivos MP3 no PC utiliza o mecanismo de comunicação inter-processos do Linux

Palavras-chave: PicBasic, Mini-ITX, Interface, PC, Norma RC5, comunicação inter-processos.

1 Introdução

O computador de bordo vem sendo aplicado com freqüência, não só em veículos pequenos, mas também em veículos de médio e grande porte.

Segundo a JcOnline (1999), os computadores de bordo funcionam por meio de sensores eletrônicos e têm como objetivo principal fornecer recursos que auxiliem na dirigibilidade e condução dos mesmos, através de consulta visual a informações específicas e gerais como temperatura, data/hora, velocidade, nível de óleo e de combustível, entre outras. Atualmente, automóveis já usam computadores de bordo capazes de controlar vários subsistemas do veículo e informar o estado geral de certos sistemas ao motorista (através de luzes no painel) e aos mecânicos através de conexões a computadores especiais.

Existem também os computadores de bordo, que focam a sua atuação em outras particularidades de um veículo, como o controle da frota de veículos de uma empresa, registrando excessos de velocidade, má condução do veículo, desgaste excessivo dos freios, com comunicação via satélite com a empresa proprietária do mesmo. Tais modelos de computador de bordo, possuem recursos de comunicação distintos dentre os quais o GPS tornou-se o mais popular. Desenvolvedores trabalham na implementação de um servidor www dentro do sistema de computador de bordo de forma totalmente embarcada.

Os avanços rápidos e contínuos na eletrônica e tecnologia da computação são um grande elemento impulsionador dos novos conceitos em controle de tráfego viário. Veículos com computadores de bordo e comunicadores poderão receber do controle de trânsito central instruções sobre o melhor caminho até o destino final. O computador de bordo também poderá informar ao computador central o seu tempo de viagem e velocidade para ser usado como parte da informação a ser processada. Em sistemas ainda mais avançados, a temporização dos semáforos será coordenada instantaneamente pelas informações recebidas dos veículos próximos (ABRAMCET, 2002).

Além de todos os detalhes abordados, acrescentam-se ainda funções de entretenimento com recursos multimídia, tais como, MP3 e DVD interligados a *displays* LCD TFTs.

O sistema proposto neste artigo visa demonstrar a integração de funcionalidades dos computadores de bordo padrões e comerciais, com um *player* de MP3. Este sistema integra a utilização de diversos conceitos e áreas de conhecimento da computação, destacando-se arquitetura de *hardware* com microcontroladores e computadores pessoais (PC), comunicação de dados serial RS232, infravermelho norma RC5 e FTP, sistema operacional Linux e integração das linguagens de programação BASIC, e C. Visa também demonstrar uma nova tecnologia de placas-mãe denominada Mini-ITX, que representa um novo formato no segmento, criado pela *VIA Technologies*, possuindo alta integração de periféricos, baixo consumo e dimensões reduzidas. Maiores detalhes podem ser consultados em Freese (2003).

Inicialmente são apresentados os conceitos teóricos mais relevantes, tais como, a arquitetura de *hardware* com informações sobre a placa-mãe Mini-ITX e o microcontrolador PIC16F877, o sistema operacional Linux com informações sobre processos e a área de comunicação de dados via infravermelho seguindo a norma RC5. A próxima seção apresenta detalhes de desenvolvimento, ilustrando a divisão de tarefas desempenhas pela interface microcontrolada e pelo PC, bem como, representações gráficas que representam a forma com a qual todos dispositivos foram interligados. Finalizando o artigo, são apresentados os resultados finais e práticos obtidos com o desenvolvimento do protótipo e a bibliografia utilizada para o embasamento teórico do artigo.

2 Arquitetura de hardware

O desenvolvimento do sistema de computador de bordo, utilizou o microcontrolador PIC16F877 e a placa-mãe Mini-ITX denominada EPIA 800 da *VIA Technologies*.

2.1 Placa-Mãe Mini-ITX

Trata-se de uma placa-mãe para plataforma embarcada x86 ultra compacta, que possui uma alta integração de periféricos (HITECHMODS, 2002). Foi desenvolvida pela *VIA Technologies*, famoso fabricante de chipsets e fabricante também dos processadores Cyrix.

Através do seu grande nível de integração de periféricos, ela ocupa 66% do espaço de uma placa-mãe padrão FlexATX e vem com um processador Cyrix C3 soldado diretamente na placa, inclusive com opção de operação sem *cooler* (VIA, 2002). A figura 1 ilustra a arquitetura de *hardware* e a tabela 1 apresenta a especificação técnica da placa-mãe Mini-ITX

O processador C3 não tem o mesmo poder de processamento dos processadores AMD ou Intel, e não possui os recursos necessários para jogos ou estações gráficas pesadas mas atende perfeitamente a diversas soluções, tais como, servidores de impressão, *players* MP3, estações de trabalho padrão (Internet, Processador de texto, sem jogos 3D...), servidores *proxy*, servidores WEB e aplicações restritamente embarcadas (TORRES, 2001).

Formato	Mini-ITX – 170x170mm – Padrão ATX
Chipsets	VIA Apollo PLE 133 – Ponte Norte VT8601A – Ponte Sul VT8231
Periféricos onboard	Video AGP2X 2D/3D (compensação DVD) – TV S-Video NTSC/PAL – USB Rede 10/100 – Serial 16C550 – Paralela EPP/ECP – Audio AC'97 – IDE
Memória	Suporta SDRAM PC100/133
Expansão	Possui um slot PCI

Tabela 1: Especificação Técnica da Placa-mãe Mini-ITX

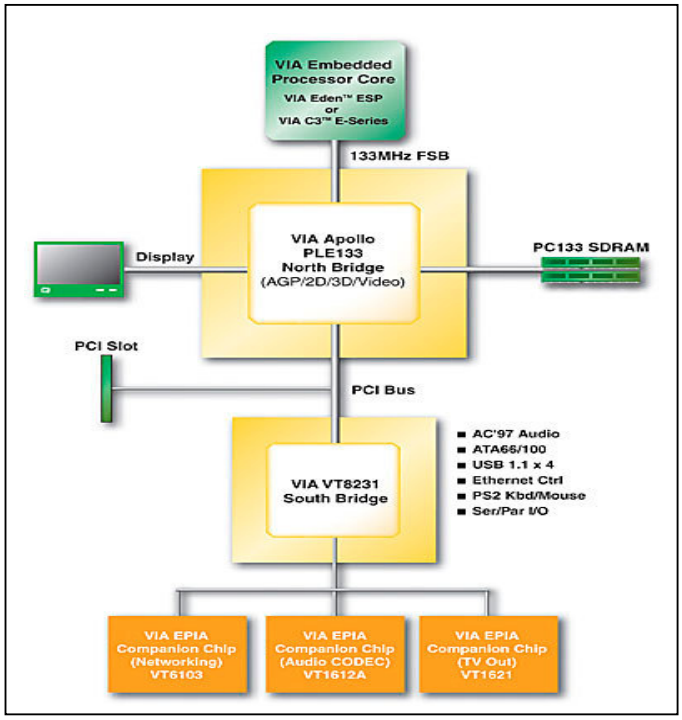


Figura 1: Arquitetura de *Hardware* Mini-ITX

2.2 Microcontrolador PIC16F877

Este microcontrolador pertence a família PIC16C7XX/F87X da Microchip. Esta família possui microcontroladores em encapsulamentos de 18 a 44 pinos com uma vasta gama de opções de integração de periféricos. Possui também instruções de 14 bits, 4 a 8 canais de A/D de 8/10 bits, capacidade de gerenciamento de interrupções, várias interfaces seriais, módulos de captura, PWM e comparadores, detecção *de Brown-Out* e pilha com até 8 níveis (MICROCHIP, 2003).

Os microcontroladores PIC16F87X possuem memória de programa FLASH e podem ser reprogramados com baixas tensões sendo ideais para aplicações de segurança e sensoreamento remoto para comando de motores e aplicações automotivas de alta velocidade de processamento (MICROCHIP, 2003). A linguagem de programação desta família é o Assembly, porém o compilador PICBasic permite a sua programação em linguagem Basic (MELAB, 2003).

3 Sistema Operacional Linux

O sistema operacional Linux é a versão mais popular do sistema operacional UNIX. Sua grande popularidade deu-se ao fato desse sistema operacional ser *open source* (TORRES, 2001).

A arquitetura do Linux é bastante estável e segura (ANUNCIAÇÃO, 1999). Trabalha em modo protegido e oferece proteção de memória para os seus aplicativos bem como multitarefa preemptiva, além de ser um sistema operacional multi-usuário. Ele foi desenvolvido para empresas, universidades, computadores pessoais (PCs) e vem sendo utilizado em larga escala em aplicações embarcadas também.

3.1 Processos UNIX

Processos são fundamentais em sistemas operacionais UNIX, controlando quase todas as atividades de um computador UNIX. É possível criar, iniciar e parar processos dentro de outros programas, assim como enviar e receber mensagens. O processo consiste numa área de memória alocada e uma *thread* que executa neste espaço seus recursos de sistema (MITCHELL, 2001).

Cada processo possui um identificador PID, uma área de código e bibliotecas que podem ser compartilhadas aumentando a eficiência de utilização da memória física, uma área de variáveis que é exclusiva de cada processo e um descritor de arquivos. Possuem também uma pilha para variáveis locais e para chamadas *call* e *return*. Estes processos são controlados por uma tabela de processos indexadas pelo PID de cada processo. Cada processo filho é iniciado por outro conhecido como processo pai (MATTHEW, 1996). A figura 2 apresenta a estrutura básica de um processo UNIX.

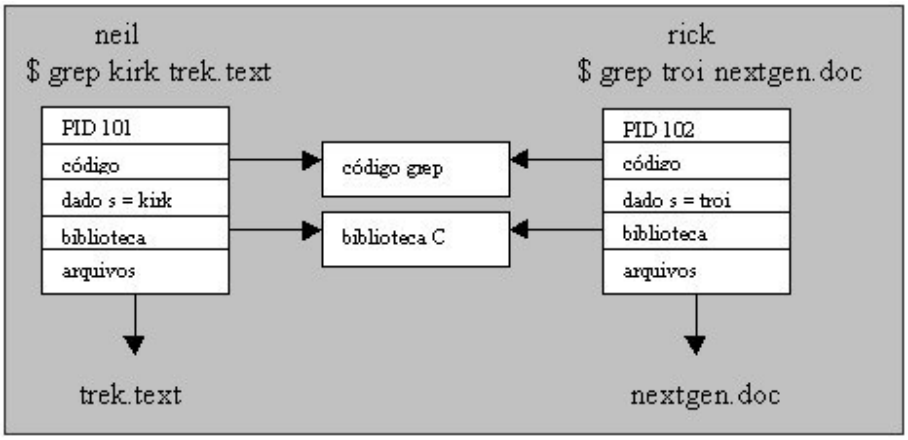


Figura 2: Estrutura de um processo UNIX

PIPE é o termo utilizado para demontrar um fluxo de dados entre dois processos, caracterizando uma comunicação entre processos. A figura 3 ilustra o redirecionamento de uma saída padrão de um processo filho para um processo pai com o fluxo de dados manipulado através da utilização de um *pipe*.

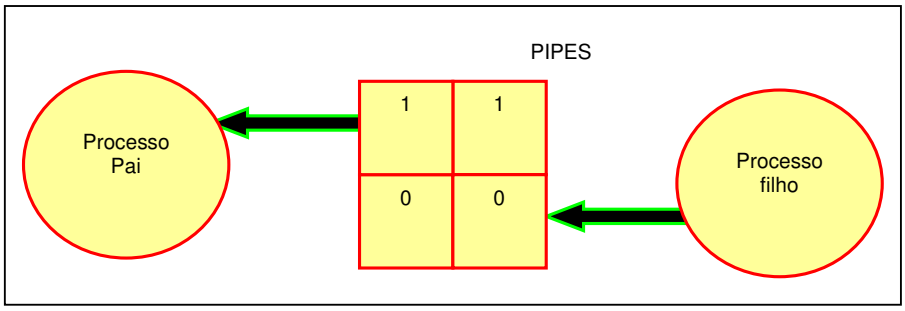


Figura 3: Redirecionamento de um saída padrão de um processo filho para um *pipe*

4 Comunicação de dados infravermelho – Norma RC5

RC5 é uma norma universal para comandos a distância por infravermelho utilizada principalmente em equipamentos de áudio, televisores, videocassetes e outros aparelhos domésticos, com uma área de alcance de aproximadamente 10m (KAINKA, 2002).

O conjunto de códigos da norma RC5 foi desenvolvido pela Phillips e possui 2048 comandos divididos em 32 grupos endereçáveis de 64 comandos cada. O código transmitido consiste de uma palavra de 14 bits, sendo eles :

- 2 bits para ajuste do nível AGC do receptor (2 start bits).O primeiro é sempre 1 e o segundo corresponde a 1 se o código de comando está entre 0-63 e 0 se está entre 64-127;
- 1 bit para controle (*check bit*) que muda de estado lógico cada vez que um botão é pressionado na unidade de comando a distância. Isto serve para indicar se o botão foi pressionado uma vez ou se continua sendo pressionado;
- 5 bits de endereço do sistema para seleção de 1 dos 32 sistemas possíveis. Isso define o tipo de aparelho que se pretende controlar;
- 6 bits de comando representando 1 dos 128 comandos possíveis. Isso define a ação que se pretende executar em um determinado aparelho (sistema) selecionado.

Tanto no endereço do sistema, quanto no comando o bit menos significativo é transmitido primeiro. A figura 4 mostra um pacote de dados norma RC5 (KAINKA, 2002).

Na norma RC5, os dados são modulados numa freqüência portadora de 30 a 40KHz.. O transmissor emite por rajadas (salvas) onde estão contidos os pacotes de dados. Cada bit transmitido tem 1,778ms de duração, enquanto que cada pulso curto tem 6,9444µs de duração e 20,8332µs de intervalo. Para uma freqüência portadora de 36KHz, cada salva curta é formada por 32 impulsos e cada salva longa, por 64 impulsos.

O pacote completo dura 24,889ms, e é sempre transmitido completamente. Se um botão do comando a distância é mantido pressionado, então o código é repetido em intervalos de 64 impulsos(113,778ms).

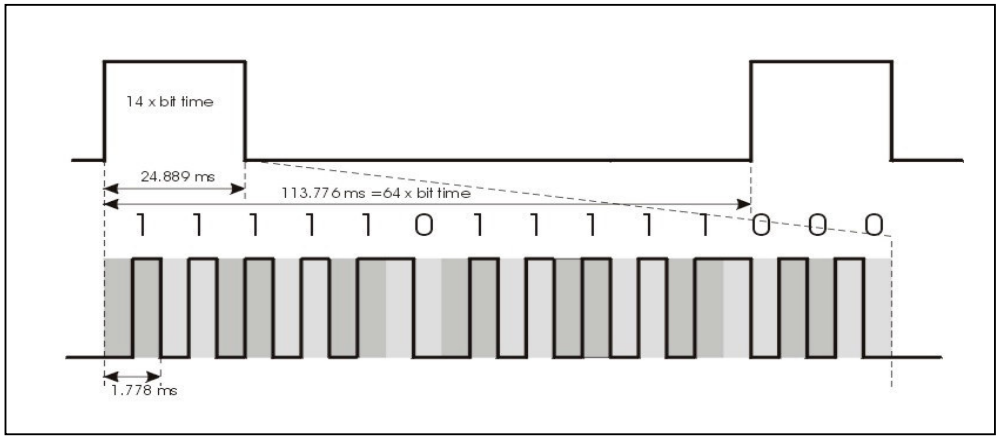


Figura 4: Pacote de dados Norma RC5 com intervalos de retransmissão

5 Protótipo Desenvolvido

Os dois tipos diferentes de *hardware* que compõem o sistema de computador de bordo, se comunicam via porta serial RS232 com taxa de transmissão de 9600 bauds. A interface microcontrolada detém controle total sobre as ações do PC, enviando serialmente comandos e recebendo dados como resposta a estes comandos. A interface também é responsável por ligar e desligar o PC, atuando diretamente sobre os pinos Pwr_On do PC, bem como, fornecendo tensão a fonte de alimentação ATX do PC, através de relés. A interface também interfere no funcionamento do sistema de som automotivo pois irá ser responsável pelo acionamento do sinal remoto do módulo, liberando a potência do mesmo, quando a função de MP3 estiver sendo utilizada. A figura 5 apresenta o diagrama de blocos do protótipo.

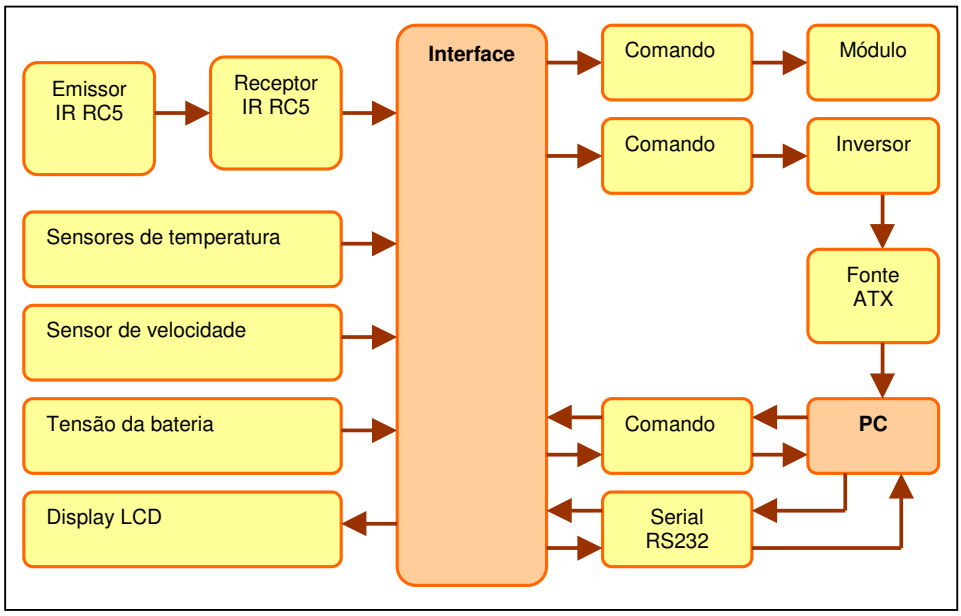


Figura 5: Diagrama de blocos do protótipo

As funcionalidades principais são :

1) Indicação de temperatura interna e externa ao veículo em °C;
2) Controle de nível de carga da bateria automotiva;
3) Indicação e controle de velocidade do veículo em Km/h, com indicação de alertas e registros de velocidades superiores a permitida via PC;
4) Indicação de data e hora atuais;
5) Reprodução de arquivos MP3 com controle de volume;
6) Interação homem/máquina através de *display* LCD 4x20;
7) Comandos emitidos via emissor infravermelho Norma RC5.

Estas funcionalidades são divididas pela interface microcontrolada e pelo PC, sendo a interface a principal responsável pela execução das mesmas, pois controla diretamente o PC. A interface recebe dados via sensor infravermelho Norma RC5 para controlar o PC. Já a comunicação entre a interface e o PC é serial, protocolo RS232. A transferência de arquivos MP3 e arquivo de excessos de velocidade do PC ocorre através do protocolo FTP, utilizando uma rede *Ethernet* ponto a ponto. As demais informações são apresentadas no *display* LCD da interface microcontrolada.

Como método de desenvolvimento, foram observadas as seguintes atividades:

a) levantamento de requisitos necessários ao desenvolvimento;
b) especificação, utilizando fluxogramas isolados para cada funcionalidade conforme as seções 5.1 e 5.2 e que integrados representam o sistema completo desenvolvido;

c) implementação, através da geração dos protótipos de *hardware* e *software* equivalentes com a especificação;

d) testes práticos, comprovando a execução do mesmo de acordo com os resultados esperados.

A especificação e implementação completa do protótipo podem ser encontradas em Freese (2003).

5.1 Interface

Dentre as principais funções do *software* da interface destacam-se :

1) Leitura de valores de temperatura através dos conversores A/Ds internos, utilizando os sensores de temperatura LM35 da *National Semiconductors*;

2) Leitura da tensão da bateria através de A/D interno. O canal A/D do PIC16F877 recebe uma tensão de 0V a 2,55V que corresponde a faixa de 0V a 17V provenientes da bateria automotiva, extraída através de um divisor de tensão variável. Através desta faixa de tensões, a tensão de referência dos canais A/D torna-se equivalente para a leitura das temperaturas, pois o protótipo lê temperaturas de 0 a 255°C (0 a 2,55V obtidos na saída do LM35);

3) Leitura de velocidade através de um pino de I/O definido, captando os sinais gerados pelo sensor de velocidade efeito *Hall* V305;

4) Interpretação dos sinais infravermelhos norma RC5 decodificando cada pacote recebido, para a leitura do sistema e do comando que deve ser executado, via interrupção. Foi utilizado o receptor infravermelho PH38C da *Siemens* e o controle remoto universal *System Link 4* da RCA como emissor;

5) Comando direto sobre o PC, substituindo os botões Pwr_On e Rst_Sw;

6) Comando direto sobre o sinal remoto do módulo de potência;

7) Comando indireto sobre a alimentação do PC, ligando e desligando o inversor de tensão;

8) Comunicação serial com o PC, com recepção através de interrupção para leitura do status do PC, informações ID3 *Tag* das músicas de MP3, tempo decorrido de execução das músicas, status de execução das músicas (*stop*,*pause*,*play*), velocidade máxima permitida sem emissão de alertas e informações de data e hora;

9) Comunicação serial com o PC, com transmissão de dados, sem interrupção, para instruções de controle sobre as músicas MP3 (*play*,*pause*,*stop*,*next*,*previous*), controle de volume, velocidade máxima permitida pelo veículo sem emissão de alertas, status de carga da bateria, desligando o PC caso necessário, bem como desligar o PC quando houver este comando via emissor infravermelho.

O componente principal da interface é o microcontrolador PIC16F877. A implementação do *software* da interface foi realizada através do compilador PICBasic Pro.

5.2 PC

Dentre as principais funções do *software* do PC destacam-se :

1) Execução de arquivos MP3 através de comunicação inter-processos. O programa utilizado foi o mpg321 que foi desenvolvido com a biblioteca MAD (*Mpeg Audio*

Decoder), que produz uma saída decodificada de 24 bits e é *freeware*. O mpg321 teve suas respectivas entrada e saída padrões redirecionadas para o programa principal;

2) Controle de volume através da criação de outro processo executando o programa aumix, redirecionando a saída padrão para o programa principal;

3) Obtenção de dados ID3 dos arquivos MP3 através de um processo executando o programa mp3info, redirecionando a saída padrão para o programa principal;

4) Registro de velocidades superiores à pré-configurada no arquivo velocmax.cfg, criando um log de velocidades no arquivo logvel.log, recebendo serialmente estas informações, através da interface;

5) Desligamento do PC através de instrução recebida serialmente, podendo ser por solicitação via emissor infravermelho ou por problemas de carga na bateria;

6) Fornecimento de data e hora através de chamadas de sistema no PC enviando as mesmas para a interface via serial;

7) Comunicação serial com a interface por *poolling*.

O *software* do PC foi desenvolvido em linguagem C para Linux utilizando o compilador gcc. Foi utilizado o sistema operacional *Linux Red Hat 7.3*.

5.3 Integração Hardware PC/Interface

A figura 6 ilustra a integração entre o *hardware* do PC e o *hardware* da Interface utilizados pelo protótipo.

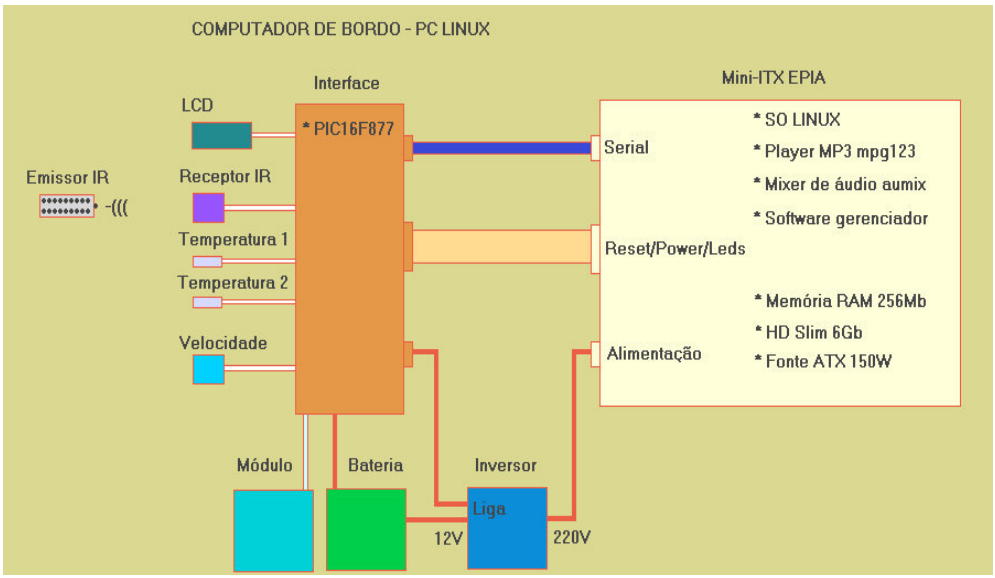


Figura 6: Integração *hardware* PC/Interface – EPIA

6 EPIA – Computador de bordo baseado em PC - Linux

O computador de bordo EPIA, é um sistema multi-funções utilizado para consulta a temperaturas interna e externa ao veículo, velocidade veicular com registro de excessos, monitoramento de carga da bateria, evitando que o veículo fique sem ignição, quando a função MP3 estiver sendo utilizada por um longo período. Além destas funções, conta também com consulta de data e hora (somente com o PC habilitado), *player* MP3 com controle de volume e playback.

Todos os comandos dos usuários são realizados via controle remoto, portanto não existem botões na interface, somente um receptor de dados infravermelho. A interface homem/máquina é feita através de um *display* LCD que apresenta as informações citadas acima, bem como informações de status dos arquivos MP3, tais como, nome da música, nome da banda, tempo decorrido, número da faixa, volume, qualidade (*stereo* ou *mono*) e status de *playback* (*play*, *pause*, *stop*). O PC possui uma fonte ATX 220V, sendo fornecida esta tensão através de um inversor de tensão de onda quadrada da Hayonik, com entrada 12V DC, provenientes da bateria do veículo.

O sistema apresenta dois modos de execução. Quando o PC está desligado são apresentadas as informações de responsabilidade da interface em tela cheia no *display* LCD. Quando o PC está ligado estas informações são apresentadas na 4ª linha do *display*, sendo as 3 linhas restantes utilizadas pelo PC para apresentar as informações sobre os arquivos MP3, identificadas acima.

A faixa de leitura de tensão da bateria extende-se de 8 a 17V, a faixa de leitura de temperaturas extende-se de 0 a 255°C, a faixa de leitura de velocidade extende-se de 1 a 180Km/h e a capacidade de armazenamento de músicas MP3 no PC é de 1300 músicas (HD 6Gb). A figura 7 ilustra o protótipo desenvolvido e a figura 8 apresenta uma tela do *display* LCD, com a reprodução de uma música MP3 apresentando suas informações nas primeiras três linhas e a velocidade veicular na última linha. A figura 9 apresenta uma tela do *display* LCD no momento em que o sistema detecta que a bateria está descarregada com o PC ligado.

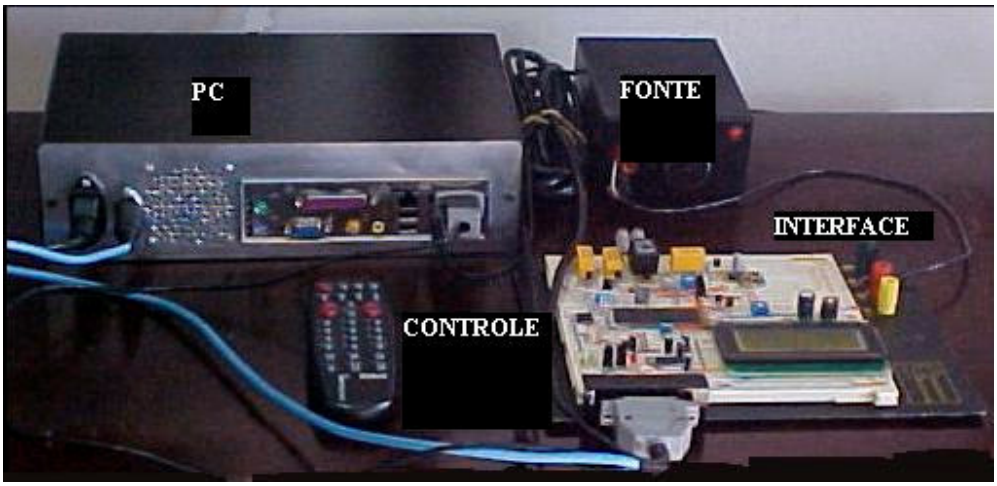


Figura 7: Computador de bordo EPIA

Figura 8: Tela de execução de arquivos MP3 com informações adicionais na 4ª linha do LCD

Figura 9: Tela de desligamento do PC por bateria descarregada

7 Conclusões

O protótipo desenvolvido, apontou para uma nova tendência em equipamentos automotivos, caracterizando um sistema de computador de bordo. Aliando a versatilidade de aplicações embarcadas dos microcontroladores com o alto poder de processamento de um placa-mãe extremamente reduzida, tivemos a oportunidade de demonstrar a interação de diversas áreas de conhecimento da computação, citando a comunicação de dados serial, a comunicação de dados infravermelho Norma RC5, o protocolo FTP, a arquitetura de *hardware*, o sistema operacional Linux, linguagem C, linguagem BASIC entre outros.

O sistema uniu funcionalidades básicas de um computador de bordo, tais como medições de temperatura, velocidade, data e hora, nível da bateria, com um recurso multimídia, já bastante difundido e que vem tornando-se um produto comercial bastante presente na área de equipamentos de som automotivo, o *player* MP3. O sistema propôs também uma funcionalidade bastante interessante para empresas, simulando a ação de um tacógrafo, através do registro de velocidades excessivas.

A utilização de uma placa-mãe para a reprodução de MP3 ao invés de um *hardware* microcontrolado com decodificador MP3, deve-se a dois fatores principalmente: o tamanho extremamente reduzido da placa-mãe utilizada, constituindo um novo padrão de placas-mãe, denominado Mini-ITX, e principalmente a possibilidade de fácil upgrade para novas tecnologias na área de compactação de áudio que tendem a se consolidar no futuro tais, como MP4, Ogg Vorbis e outras que surgirão em meio ao avanço tecnológico cada vez mais constante e ágil.

7.1 Sugestões para trabalhos futuros

Criação de uma nova funcionalidade, para medição de consumo de combustível a cada abastecimento em tempo real com registro de operações.

Implementação de rotinas gráficas para o limitado *display* LCD padrão HD44738, aumentando o nível de estética de apresentação de dados, juntamente com um equalizador gráfico para o *player* MP3.

Implementação de um sistema de direção inteligente, que funcionaria com um detector de saída de pista do veículo, caso por exemplo o motorista pegasse no sono, utilizando o poder de processamento da placa-mãe EPIA Mini-ITX.

Implementação de um sistema de GPS, com a placa-mãe Mini-ITX.

8 Referências

ABRAMCET. **Computador de bordo,** São Paulo, 2003. Apresenta informações sobre novas tendências para os computadores de bordo. Disponível em: <http://www.abramcet.com.br/home/old/hist_conceitos.shtml>. Acesso em: 6 fev. 2003

ANUNCIACAO, Heverton Silva. **Linux** – guia prático em português**.** São Paulo: Érica, 1999.

CANTU, Marcos. **Delphi 5.0** – a bíblia. São Paulo: Makron Books, 2000.

CANZIAN, Edmur. Comunicação Serial – **RS232**. São Paulo: Editora da Escola Técnica CNZ de Cotia, 2002.

DANESH, Arman. **Dominando Linux** – a bíblia**.** São Paulo: Makron Books, 2000.

FIP. **Sensores de velocidade**, São Paulo, 2002. Ilustra funcionamento e instalação dos sensores de velocidades comercializados pelo mesmo. Disponível em: <http://www.fip.com.br/Sensores/Sensor%20de%20Velocidade.pdf>. Acesso em: 9 mar. 2003.

FREESE, Cristiano. **Protótipo de um computador de bordo baseado em PC-Linux.** Blumenau, 2003. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) Universidade Regional de Blumenau.

HITECHMODS. **VIA EPIA-800** Mini-ITX Motherboard Review, Flórida, 2002. Apresenta uma revisão completa sobre a placa-mãe EPIA 800 Mini-ITX. Disponível em: < http://www.hitechmods.com/reviews/motherboards/VIA_EPIA/via_epia.shtml >. Acesso em: 20 out. 2002.

JCONLINE. **Computadores de bordo vs check control**, Recife, 1999. Apresenta particularidades dos dois tipos de sistema. Disponível em: < http://www2.uol.com.br/JC/_1999/1903/vc1403b.htm>. Acesso em: 12 fev. 2003.

KAINKA, Brian. Emissor/receptor IV para PC. **ELEKTOR**, São Paulo, ano 1, n. 7, p. 6-11, out. 2002.

KERNIGHAN, Brian W.; RITCHIE, Dennis M. **C: A linguagem de programação**. Rio de Janeiro: Campus, 1990.

The GCC Team. **GCC Home Page – GNU Project**, Boston, 2003. Site oficial do compilador GCC. Disponível em: < http://gcc.gnu.org/>. Acesso em: 30 mai. 2003.

NATIONAL SEMICONDUTOR. **Analog and interface products databook**. California: National, 2002.

MATTHEW, Neil; STONES, Richard. **Beginning Linux programming**. Birmingham: Wrox, 1996.

MELAB MicroEngineering Labs Corporation. **PicBasic** compiler, Califórnia, 2003. Apresenta manuais e bibliotecas existentes para utilização do compilador PicBasic. Disponível em: <http://www.melabs.com/resources/index.htm>. Acesso em: 01 mar. 2003.

MITCHELL, Mark; OLDHAM, Jeffrey; SAMUEL, Alex. **Advanced Linux Programming**. Indiana: New Riders, 2001.

MICROCHIP. **PIC16F877** device, Arizona, 2001. Apresenta informações técnicas sobre o microcontrolador PIC16F877 e exemplos de aplicação prática. Disponível em: <http://www.microchip.com/1010/pline/picmicro/category/embctrl/14kbytes/devices/16f877/index.htm>. Acesso em: 10 jan. 2003.

SCHILDT, Herbert. **C completo e total**. 2. ed. São Paulo: Makron Books, 1990.

STEVENS, W. Richard. **Advanced programming in the UNIX environment**. Boston: Addison-Wesley, 1993.

TORRES, Gabriel **Hardware** – curso completo. 4. ed. Rio de Janeiro: Axcel Books, 2001.

VIA. **EPIA Mini-ITX user's manual**. Newark: Via VPSD, 2002.

WINSTAR Displays. **WH2004** product, Taipei, 2003. Apresenta dados técnicos do display LCD WH2004. Disponível em: < http://www.winstar.com.tw>. Acesso em: 15 mar. 2003.

Avaliação e Adequabilidade de Benchmark Paralelos Modernos

Omar Andres Carmona Cortes
Regina H. C. Santana
Marcos J. Santana
Universidade de São Paulo
(Instituto de Ciências Matemáticas e de Computação)
{ocortes,rcs,mjs}@icmc.usp.br

Resumo. O objetivo deste artigo é mostrar uma análise dos principais benchmarks paralelos disponíveis gratuitamente para avaliação de desempenho de arquiteturas paralelas. Os benchmarks paralelos são avaliados quanto a escalabilidade, portabilidade, instalação, execução e documentação. Este trabalho visa também forcer informações úteis sobre a adequabilidade dos benchmarks paralelos para usuários que necessitam avaliar o desempenho de suas ferramentas e/ou arquiteturas paralelas de maneira confiável.

Palavras-chave: Avaliação de Desempenho, Benchmarks Paralelos, Computação Paralela, Adequabilidade

1 Introdução

Analisar o desempenho de máquinas paralelas é uma tarefa relativamente complexa devido ao não determinismo das aplicações executadas nessas máquinas, ou seja, nada garante que a mesma tarefa seja executada da mesma forma em instantes consecutivos de tempo.

Mesmo a avaliação de arquiteturas seqüenciais não é trivial. A definição de métricas adequadas e de formas confiáveis para obtenção dessas métricas é essencial para uma avaliação de desempenho confiável. Nesse âmbito surgiram os benchmarks [24] como uma poderosa ferramenta. Os benchmarks são capazes de testar recursos de arquiteturas paralelas tais como: custo de comunicação, sincronização de barreiras, capacidade matemática do processador, capacidade de vetorização, etc.

Dado esse contexto, o objetivo deste trabalho é oferecer aos usuários de computação paralela uma análise sobre os principais benchmarks paralelos disponíveis. Para fornecer subsídios a um usuário que deseje analisar o desempenho de um sistema [1] qualquer é feita uma análise através dos seguintes fatores: facilidade de instalação e compilação; escalabilidade; documentação e execução. Cada um desses fatores é detalhado na Seção 4.1.

Para cumprir sua tarefa este artigo está dividido da seguinte forma: a Seção 2 apresenta os conceitos básicos sobre benchmarks paralelos; a Seção 3 mostra como os benchmarks estão sendo utilizados nos dias atuais. A Seção 4 resume as principais características dos benchmarks paralelos utilizados juntamente com uma análise quanto a escalabilidade, dificuldades de execução, instalação e documentação; a penúltima seção é responsável por mostrar as conclusões em relação a utilização de benchmarks paralelos e a última seção apresenta as referências bibliográficas.

[1] Entende-se por sistema qualquer recurso de hardware ou software paralelo que tenha sido desenvolvido.

2 Conceitos Básicos de Benchmarks Paralelos

A Intel Corporation [27], define um benchmark como um programa o qual mede o desempenho de um computador, de uma parte do mesmo ou de um outro programa os quais executam sempre a mesma tarefa periodicamente. Dessa forma, um benchmark pode refletir o comportamento de um recurso computacional, seja este recurso um programa, um subsistema ou uma arquitetura completa.

A verificação do desempenho de um único componente do sistema não permite uma avaliação precisa do desempenho global, embora uma vez conhecido o desempenho global a avaliação de um único componente pode mostrar o seu impacto dentro de um sistema como um todo.

Nesse contexto, para avaliar os diferentes aspectos de um sistema paralelo os benchmarks podem ser divididos por camadas [16]. Nessa abordagem o benchmark é executado nos seguintes níveis de complexidade:

- **Baixo Nível** - Conhecido dessa forma por avaliar basicamente as propriedades físicas da máquina, tais como vazão de dados, tempo necessário para sincronização entre processos, custo de barreiras, etc. Deve-se perceber que essas mesma informações podem também ser utilizadas para avaliar o desempenho compiladores e bibliotecas para computação paralela.
- **Núcleo** - Executa porções de código que geralmente são utilizadas em simulações científicas, tais como resoluções de integrais, resolução de equações diferenciais, multiplicação de matrizes, resolução de sistemas lineares, etc. Geralmente este nível é utilizado para verificar a capacidade numérica de um ou mais processadores ou de um compilador.
- **Aplicações Compactas** - Executa simulações complexas que muitas vezes utilizam o código que está em nível de núcleo. Dentre as simulações mais utilizadas nesta camada estão: a cromo-dinâmica, a termodinâmica, a meteorologia, a mecânica de fluidos, e a simulação de dispositivos eletrônicos entre outras. Também chamado de nível de simulação. Com esta camada pode-se medir o desempenho de uma arquitetura paralela em uma possível situação real.

Os benchmarks paralelos ainda podem ser considerados como *suite*, isto é, um pacote que é formado por benchmarks de todas as camadas.

A próxima seção apresenta o estado da arte na utilização benchmarks paralelos e as linguagens mais utilizadas no desenvolvimento.

3 Estado da Arte e Trabalhos Correlatos

A grande maioria dos trabalhos utilizam benchmarks paralelos sob dois aspectos. O primeiro diz respeito a utilização de benchmarks paralelos para avaliar algum tipo de produto desenvolvido, seja este produto uma biblioteca, um compilador ou uma arquitetura. O segundo aspecto trata do desenvolvimento de benchmarks paralelos abordando especialmente a linguagem de desenvolvimento e as métricas utilizadas.

Atualmente dentre as linguagens disponíveis, a linguagem OpenMP [18] tem estimulado o desenvolvimento e a migração de diversos benchmarks paralelos para esta linguagem.

Em [2] apresenta-se um novo benchmark do tipo *suite* desenvolvido em OpenMP e são efetuados alguns testes de portabilidade de código. Já no trabalho de Kusano [29], avalia-se o desempenho de um compilador para OpenMP. Lin et al. [30] apresenta uma comparação das linguagens ZPL [43] e HPF [26] tendo como referência o NAS Parallel Benchmark.

Outra área que tem despertado interesse é a utilização e avaliação de estruturas de dados derivadas do MPI [32]. Essas estruturas permitem criar tipos de dados especiais a serem utilizados na comunicação entre processos. Nos trabalhos [31, 37] avalia-se o desempenho desse tipo de estrutura de dados em diferentes arquiteturas.

A utilização de benchmarks de baixo nível também tem sido bastante discutida. O trabalho de Grove [22] mostra o desenvolvimento de um novo benchmark de baixo nível e faz uma comparação com outros benchmarks do mesmo tipo. Já no trabalho de Dimitrov et al. [15], utiliza benchmarks de baixo nível juntamente com o NAS Parallel Benchmark para avaliar o impacto da latência em aplicações paralelas.

Com relação a utilização de benchmarks para avaliação de um produto, sem duvida nenhuma, o NAS Parallel Benchmarks é o mais utilizado. Em Chamberlain et al. [11], é feita uma análise de um dos benchmarks que compõem o NAS (MG) através de diversas arquiteturas. Takahashi et al. [44] utiliza o NAS para avaliar uma camada de *Middleware* por ele desenvolvida. Outros trabalhos nessa mesma linha são: [29, 10, 7].

Mostrados os conceitos básicos e as formas mais comuns de utilização dos benchmarks, a próxima seção apresenta as características e a análise de alguns benchmarks paralelos.

4 Descrição e Análise de Benchmarks Paralelos

Na escolha dos benchmarks paralelos a serem analisados neste artigo foram levados em consideração a utilização gratuita (benchmarks livres) e os benchmarks citados em trabalhos relativamente recentes.

A próxima seção aborda a metodologia utilizada para analisar cada um dos benchmarks escolhidos.

4.1 Metodologia

A execução dos benchmarks para análise deu-se em dois ambientes. O primeiro ambiente é um IBM-SP2 com três nós e sistema operacional Unix AIX. O segundo é um sistema distribuído com 4 máquinas do tipo PC com sistema operacional Linux. Os detalhes dos ambientes são omitidos, pois o desempenho dos mesmos não esta sendo avaliado.

A avaliação de cada benchmark será feita através de uma tabela levando em consideração os seguintes fatores: facilidade de instalação e compilação, escalabilidade, execução e documentação. A escolha de cada um dos fatores deu-se pelos seguintes motivos:

- **Facilidade de instalação e compilação** - Alguns benchmarks paralelos trazem arquivos de configuração pré-definidos para compilação em determinadas arquiteturas. Esses arquivos facilitam o trabalho de instalação, configuração e compilação do benchmark, especialmente quando se trata de usuários não familiarizados com a arquitetura que está sendo avaliada. Entretanto, se esses arquivos forem confusos e com o código mal documentado, a configuração e a compilação pode ser consideradas complexas. De maneira geral, um benchmark paralelo com arquivos de configuração bem estruturados e bem documentados não geram nenhum tipo de problema na compilação. Todavia,

alguns problemas na compilação podem ocorrer como pode ser visto nos trabalhos de Cortes [12] e Satake [41].

- **Escalabilidade** - A *Escalabilidade* é a capacidade que um benchmarks paralelo tem de adequar o tamanho da estrutura de dados utilizada no algoritmo à capacidade do sistema que está sendo avaliado. O tamanho da(s) estrutura(s) de dados é comumente chamado de *tamanho do problema*. A escalabilidade é importante porque a medida que as arquiteturas evoluem o benchmark deve ser capaz de solucionar problemas cada vez maiores. Caso contrário, o benchmarks se tornará rapidamente obsoleto perdendo completamente sua utilidade.

- **Documentação** - A documentação em um benchmark paralelo é um item extremamente importante, pois a mesma deve fornecer informações sobre a instalação, as regras de execução do benchmark, como alterar o tamanho do problema (estrutura de dados) que está sendo solucionado e os ajustes permitidos para que na avaliação não haja tendências a favorecer uma determinada arquitetura.

- **Execução** - Muitas vezes uma compilação feita com sucesso não garante uma execução sem problemas. Dependendo da arquitetura podem ocorrer especialmente problemas de memória. Em alguns casos podem ocorrer também pequenos problemas no código da versão disponibilizada.

Para cada um dos fatores recém mostrados é atribuído um conceito que segue as seguintes regras de atribuição:

- **Ruim** - Quando o fator analisado não apresenta o item que está sendo avaliado ou os problemas gerados tornaram inviável a utilização do benchmark. Por exemplo: falta de documentação, impossibilidade de compilação, etc.

- **Regular** - Este conceito é dado quando os problemas apresentados pelo benchmark começam a afetar a análise do sistema em questão. Neste caso algumas conclusões sobre o sistema em avaliação ainda podem ser tiradas.

- **Bom** - Para benchmarks com pequenos problemas que não perturbam a análise do sistema como um todo.

- **Excelente** - Fornecido a benchmarks que não apresentaram nenhuma espécie de problema.

Com a metodologia bem definida a seguir são apresentadas as características dos benchmarks utilizados, trabalhos recentes que os utilizaram e as respectivas análises.

4.2 Benchmarks Paralelos Utilizados

4.2.1 NAS Parallel Benchmark

O NAS foi criado pelo NASA Ames Laboratory que fica localizado na California. O NAS surgiu devido à necessidade de avaliar sistemas paralelos de alto desempenho que não fossem vetoriais [3, 46], além da necessidade de verificar a portabilidade das máquinas, característica que é de fundamental importância. Atualmente está na versão 2.4 que foi apresentada

em outubro de 2002 [46]. Este benchmark é construído com oito algoritmos (denominados por duas letras), onde cinco são do tipo núcleo (IS,EP,MG,CG,FT) e três aplicações compactas(LU,SP,BT).

Os algoritmos de núcleo trabalham com o algoritmo de monte carlo (EP), solução de equações de Poisson (MG), resolução de sistemas lineares através do método de gradiente conjugado (CG), resolução de transformadas rápidas de Fourier (FT) e ordenação de inteiros (IS). As aplicações compactas são todas baseadas em CFD (Computer Fluid Dynamics - Dinâmica de Fluídos por Computador) e utilizam três métodos para solucionar equações. O primeiro método é o LU que fatora uma matriz regular-esparsa até sua solução utilizando blocos de 5x5. O segundo método utilizado é o SP, que faz a multiplicação de sistemas independentes não-diagonais dominantes e resolve equações escalares pentadiagonais. O terceiro e último método utilizado é o BT que é muito semelhante ao SP sendo que a diferença está na quantidade de dados que são transmitidos entre os processadores [3]. Todos os benchmarks foram desenvolvidos em Fortran 77 com exceção do IS que foi implementado em C.

Apesar da versão 2.3 ter sido desenvolvida em 1996, essa versão do NAS foi bastante utilizada até recentemente como pode ser visto nos trabalhos de [9], [4] e [22]. Esse motivo incentivou o desenvolvimento da versão 2.4 (Fortran + MPI). Versões em outras linguagens tais como HPF [34, 19], ZPL [34], OpenMP [28] e Java [20] tem sido desenvolvidas e disponibilizadas em um pacote denominado NAS 3.0 (com exceção da versão em linguagem ZPL). Essa modernização no código dá uma maior aceitação do benchmark nas comunidades de avaliação de desempenho, desenvolvedores de aplicações paralelas e fabricantes de máquinas paralelas e de alto desempenho.

Análise

A instalação e compilação do NAS no IBM-SP2 foi de extrema simplicidade o que faz com que a portabilidade de código seja excelente. A escalabilidade é boa apenas por ser limitada às classes pré-definidas [2]. A documentação é excelente, pois apresenta de forma clara todos os passos que devem ser seguidos para a execução. Na Tabela 1 são mostrados os resultados obtidos na análise do NAS.

Tabela 1: Análise do NAS Parallel Benchmark

Arquitetura	**Instalação e Compilação**	**Escalabilidade**	**Execução**	**Documentação**
IBM-SP2	Excelente	Bom	Excelente	Excelente
Cluster	Bom	Bom	Excelente	Bom

Segundo a documentação, o NAS não é recomendado para a execução em cluster de PC´s, pois provavelmente não funcionaria. O fato do benchmark ter sido compilado e executado leva um ponto positivo para o NAS embora a escalabilidade dos problemas foi afetada devido à limitação de memória das máquinas do tipo PC. A instalação e execução são consideradas boas, pois o benchmark FT não funcionou na arquitetura de *Cluster* e os

[2] O NAS pode ser executado em 5 classes diferentes S,W,A,B,C,D. Cada uma delas apresenta um tamanho de problema diferente, sendo S a menor tamanho e D a maior (utilizada para avaliar arquiteturas em *Grid Computing*) .

benchmarks SP e MG exigiram uma mudança no arquivo de compilação Makefile que não está documentada.

4.2.2 Pallas MPI Benchmark (PMB)

O Pallas MPI Benchmark foi desenvolvido em 1999 em linguagem C (padrão ANSI) com MPI [35]. É um benchmark de baixo nível. Apresenta códigos para avaliar o desempenho através de rotinas de comunicação ponto-a-ponto, rotinas de comunicação coletiva e custo na utilização de barreiras. Uma parte do PMB chamada de EFF, formada apenas pelas rotinas de comunicação ponto-a-ponto, pode ser utilizada separadamente. Os benchmarks encontrados no EFF medem apenas o tempo de inicialização e a vazão de dados (*bandwidth*).

O trabalho de [17] utiliza o Pallas MPI para avaliar o desempenho do MPICH em diversas arquiteturas: cluster de estações de trabalho utilizando redes Ethernet (Gigabit e Fast), Myrinet e um IBM-SP2. O Pallas também é citado no trabalho de [40].

Análise

Por ser um benchmark específico, ou seja, apresenta código para medir apenas as características de baixo nível da arquitetura, o mesmo não apresentou nenhum problema de compilação ou de portabilidade entre o IBM-SP2 e o cluster de PC's, embora dois dos benchmarks não funcionaram no cluster de PC's devido a problemas de memória. Dessa forma atribuiu-se a nota de boa para execução no cluster de PC's. A escalabilidade é excelente e fácil de ser manipulada, pois é feita via um arquivo de configuração. A Tabela 2 mostra o resutlado da avaliação do PMB.

Tabela 2: Análise do Pallas MPI Benchmark

Arquitetura	**Instalação e Compilação**	**Escalabilidade**	**Execução**	**Documentação**
IBM-SP2	Excelente	Excelente	Excelente	Excelente
Cluster	Excelente	Excelente	Bom	Excelente

A documentação é bem clara apresentando passo a passo a instalação, a compilação e a execução dos códigos, além de possuir na própria documentação detalhes de como cada um dos benchmarks realiza a comunicação e a sincronização.

4.2.3 LLCBench

Este benchmark é um dos mais recentes, sua última versão é de julho de 2000 e seu código também foi desenvolvido em C [42]. É formado por três benchmarks independentes: MPBench, CacheBench e BlasBench. Uma característica muito interessante deste benchmark é que ele próprio gera gráficos de desempenho através de uma interface com o software GNU Plot. Como só o MPBench é paralelo somente ele será analisado.

O MPBench é de baixo nível e foi desenvolvido com o objetivo principal verificar o desempenho de implementações do MPI em arquiteturas paralelas, podendo ser utilizado também para avaliar o sistema de comunicação ponto-a-ponto e de comunicação coletiva.

Os benchmarks de baixo nível do MPBench podem ser extremamente úteis para verificar o comportamento da camada de passagem de mensagem de implementações MPI. Como o código é pequeno e simples facilitando a portabilidade de código. Ele pode ser alterado, por exemplo, para realizar a comparação entre implementações MPI e PVM [5].

Em [22] o MPBench é comparado a outros benchmarks tais como: mpptest e SKaMPI. Já no trabalho de [40] o MPBench é citado como um benchmark que pode avaliar algumas funções de um sistema de comunicação em MPI.

Análise

O MPBench não apresentou nenhum problema de instalação nem de compilação. A documentação é excelente apresentando todos os passos necessários para instalação, compilação e execução. A escalabilidade é excelente e feita via arquivos de configuração. Este benchmark ganha pontos extras por dar resultados também na forma de gráficos automaticamente gerados. A documentação apresenta detalhes sobre a implementação de cada um dos benchmarks. A Tabela 3 mostra o resultado da avaliação no MPBench.

Tabela 3: Análise do MPBench

Arquitetura	**Instalação e Compilação**	**Escalabilidade**	**Execução**	**Documentação**
IBM-SP2	Excelente	Excelente	Excelente	Excelente
Cluster	Excelente	Bom	Excelente	Bom

A execução no MPBench exige uma interatividade maior com o sistema que está sendo avaliado devido à forma de como os resultados são apresentados (arquivos de texto e gráficos), por esse motivo a execução no IBM-SP2 foi um pouco prejudicada já que este sistema trabalha na forma de batch. Embora esse pequeno problema tenha ocorrido, o conceito do fator *Execução* não foi prejudicado pois o benchmark apresenta os resultados também na forma de texto.

Para a arquitetura de cluster o conceito de *Execução* e *Escalabilidade* caem para *Bom* por apresentar problemas durante a execução ao serem utilizados os tamanhos das estruturas de dados padrão. O benchmark também não apresentou problemas de portabilidade de código ao ser portado do IBM-SP2 para o *Cluster*. A instalação e a compilação deu-se sem nenhum problema em ambas arquiteturas.

4.2.4 Genesis Low Level

Este benchmark foi implementado em MPI no ano de 1998 sendo incorporado ao projeto Gênesis [3]. Pode ser considerado como a única atualização do Gênesis 3.0 [1] até hoje. Possui códigos apenas para avaliar a comunicação ponto-a-ponto com algumas variações nas estruturas de dados que fazem parte da mensagem. Este benchmark foi desenvolvido especialmente para testar as capacidades de comunicação de uma arquitetura paralela utilizando MPI. Assim como em outros benchmarks de baixo nível também se pode testar as implementações de MPI disponíveis no mercado.

[3] O Genesis é um benchmark paralelos do tipo *suite* e não está disponível para *download* desde 1997.

O Gênesis Low Level possui uma característica interessante, que não está presente a maioria dos benchmarks semelhantes, que é a utilização de um tipo intrínseco do MPI para instanciação de variáveis que é o MPI_TYPE_VECTOR. Essa característica permite que usuários possam testar a sobrecarga na utilização deste tipo de variável. O trabalho de [23] trata do desenvolvimento do Genesis Low Level.

Análise

A instalação e compilação do Genesis Low Level foi simples e fácil nas duas arquiteturas paralelas onde ele foi testado. A Tabela 4 mostra o resultado da análise de Gênesis Low Level. A documentação é pobre e incompleta, não apresenta nem informações sobre a escalabilidade nem sobre as regras de execução fazendo com que os respectivos conceitos caiam para *Ruim*.

Tabela 4: Análise do Genesis Low Level

Arquitetura	**Instalação e Compilação**	**Escalabilidade**	**Execução**	**Documentação**
IBM-SP2	Excelente	Excelente	Regular	Ruim
Cluster	Excelente	Excelente	Regular	Ruim

A execução foi considerada regular por estarem faltando arquivos para um dos benchmarks, esse problema foi solucionado copiando-se o arquivo de um benchmark semelhante no mesmo pacote. Dependendo da experiência do usuário com benchmarks a solução encontrada para o problema de falta de arquivos não é trivial. A escalabilidade apesar de não documentada apresenta os arquivos de configuração de forma bem intuitiva fazendo com que seja mantido o respectivo conceito.

4.2.5 Special Karlsruher MPI Benchmark - SKaMPI

O SKaMPI foi desenvolvido em C com MPI com o objetivo de medir o desempenho das diferentes implementações do MPI. Outro objetivo deste benchmark é também criar uma base de dados contendo as informações sobre os resultados encontrados em diversas arquiteturas permitindo que: usuários decidam qual a plataforma mais eficiente para desenvolverem suas aplicações e descrever o comportamento do MPI em supercomputadores [25]. A última versão disponível para download é a 3.0 e apresenta as seguintes características [38]:

- **Limitação de tempo** - é permitido ao usuário limitar o tempo máximo para a execução de um laço.
- **Operações coletivas** - todas as operações coletivas do MPI são testadas.
- **Temporização por nó** - os tempos coletados não são apenas os tempos que cada teste leva para ser executado, mas também apresentas as informações de temporização em cada processo envolvido.
- **Resultados configuráveis** - o usuário pode determinar dentre todos os resultados que podem ser apresentados somente os que são de maior relevância para o teste.

Dos benchmarks de baixo nível, o SKaMPI é o que mais tem sido utilizado. Alguns dos trabalhos que tratam desse benchmark são: [38], [8], [22], [39], [36] e [40].

Análise

O SKaMPI apresenta características interessantes que são a limitação de tempo, a temporização por nó e a possibilidade de configurar os resultados sem alteração no código. Essas características, que não estão presentes nos demais benchmarks com a mesma função, dão uma grande flexibilidade na utilização do SKaMPI. A Tabela 5 mostra o resultado da análise do SKaMPI.

Tabela 5: Análise do Special Karlsruher MPI Benchmark

Arquitetura	**Instalação e Compilação**	**Escalabilidade**	**Execução**	**Documentação**
IBM-SP2	Excelente	Excelente	Excelente	Boa
Cluster	Excelente	Excelente	Boa	Boa

As avaliações são feitas todas no mesmo código. Fato este que levou tanto a compilações quanto a execução de forma simples e rápida. Embora a execução tenha sido simples ela é considerada boa, pois a execução é abortada quando o benchmark é executado com a configuração padrão na arquitetura de Cluster. A documentação é bem detalhada, mas recebe uma avaliação de boa, pois a configuração dos relatórios de resultados exigem um estudo aprofundado da documentação e um a familiaridade com a linguagem PERL.

No que tange a portabilidade de uma arquitetura para outra, o SKaMPI não apresentou absolutamente nenhum problema. O principal motivo é por ser um benchmark de código único, ou seja, todos os testes estão no mesmo arquivo de programa. Esse fator fornece ao benchmark uma extrema facilidade na compilação do código.

4.3 Mpptest

O Mpptest [21] foi criado pela Argonnor National Laboratory em 1999. O benchmark é de baixo nível e foi desenvolvido em linguagem C. Seu objetivo é avaliar a comunicação ponto-a-ponto síncrona e assíncrona, assim como executar avaliações em um grupo de operações coletivas.

Basicamente são criados dois programas. Um denominado *mpptest* que faz a avaliação das operações ponto-a-ponto e coletivas em função do tamanho da mensagem. O segundo, chamado *goptest* faz a mesma operação que o mpptest em função do número de processos executados. Scripts permitem escolher os testes que irão gerar dados para gráficos do Gnuplot.

Análise

Uma das características mais interessantes do mpptest é a capacidade de determinar automaticamente o tamanho das mensagens que serão trocadas. Outro ponto que deve ser destacado é a interface com o Gnuplot, que facilita a interpretação dos dados de saída.

Tabela 6: Análise do mpptest

Arquitetura	Instalação e Compilação	Escalabilidade	Execução	Documentação
IBM-SP2	Excelente	Excelente	Excelente	Excelente
Cluster	Excelente	Excelente	Excelente	Excelente

Nos dois ambientes utilizados o mpptest não apresentou nenhum problema de instalação. A escalabilidade é excelente por ser automática facilitando o trabalho do usuário. O processo de execução não apresentou nenhum tipo de contratempo. Para finalizar, a documentação é suficiente para instalação, compilação configuração e execução.

5 Adequabilidade

Observa-se através da Tabela 7 que todos os benchmarks que avaliam somente o sistema de comunicação estão implementados em C. Isso se deve ao fato da linguagem C ser mais poderosa que a linguagem Fortran 77 em alguns aspectos, tais como: alocação dinâmica de memória, tratamento de entrada/saída, etc.

Tabela 7: Adequabilidade dos benchmarks paralelos

Nome	Versão e Ano	Linguagens	Adequabilidade
NAS	2.4 2002	Fortran-MPI, HPF, ZPL e OpenMP	Faz simulação de CFD, testa a capacidade matemática, se o algoritmo usado é conhecido pelo usuário pode-se avaliar o sistema de comunicação.
PMB	1999	C-MPI	Avalia sistemas de comunicação através de rotinas de comunicação simples e coletivas.
LLC	2000	C e C-MPI	Composto por três benchmarks. Excelente para avaliar o sistema de comunicação e a capacidade matemática da biblioteca Blas.
Genesis Low Level	1999	C-MPI	Avalia o sistema de comunicação através de rotinas ponto-a-ponto.
SKaMPI	2.0 1999	C-MPI	Avalia o sistema de comunicação. Possui características interessantes tais como a configuração de tempo e de dados de saída.
mpptest	1999	C-MPI	Avalia o sistema de comunicação síncrona e assíncrona. Possui características interessantes tais como a definição automática do tamanho da mensagem na comunicação e geração de dados para gráficos.

Os benchmarks de baixo nível são úteis devido a grande quantidade de interconexões disponíveis nas arquiteturas paralelas. De maneira geral, esse tipo de benchmark é fácil de instalar e de utilizar. Os que precisam de ferramentas extras para apresentação dos dados e geração de gráficos exigem um pouco mais de conhecimento do sistema, mas apenas para gerar os relatórios de resultados, sendo que na instalação e na execução mantém-se a facilidade de uso. O SKaMPI, mpptest e o MPIBench são exemplos de benchmarks de baixo nível que exigem mais conhecimento para gerar os relatórios de resultados.

Os benchmarks de núcleo e de aplicação são importantes devido a quantidade de aplicações científicas utilizadas, as quais geralmente precisam de um grande esforço computacional para serem executadas . Esse tipo de benchmark é capaz de avaliar a capacidade matemática de uma arquitetura juntamente com o sistema de comunicação, já que os processos paralelos devem trocar informações entre si. Nota-se que todos os benchmarks que fazem avaliações através de algoritmos numéricos estão desenvolvidos em Fortran devido a excelente capacidade numérica da linguagem.

Dentre os benchmarks paralelos abordados, o mais atualizado é o NAS como pode ser visto nos trabalhos de [29, 15, 10, 11, 7]. Os principais motivos do NAS ser o mais utilizado são: não tem custo de aquisição, possui o código aberto, possui uma larga aceitação na comunidade científica e é relativamente fácil de usar. Por ser o mais utilizado possui versões em HPF [34, 19], ZPL [34], OpenMP [28] e recentemente em Java [20].

6 Conclusões

Este trabalho apresentou uma avaliação dos principais benchmarks que podem ser facilmente encontrados para utilização gratuita. As pesquisas nessa área vêm crescendo, sendo que o principal alvo dos benchmarks paralelos é a computação paralela científica pois, a computações científica exige cada vez mais recursos computacionais.

Nesse contexto, destaca-se a linguagem Fortran que provê alto desempenho numérico [14, 45, 33, 13], sendo essa a razão da maioria dos benchmarks estarem escritos em Fortran com suas extensões paralelas. Observa-se que a partir de 1998 diversas versões em C foram desenvolvidas (mpptest, MPIBench, SKaMPI e LLCBench entre outros), mas todos esses benchmarks são de baixo nível não exigindo poder matemático do processador.

De modo geral a maioria dos benchmarks paralelos utiliza bibliotecas de passagem de mensagem tais como o PVM ou MPI que permitem uma mudança de plataforma praticamente sem nenhuma mudança de código. Outras linguagens mais modernas estão começando a ser bem aceitas pela comunidade de benchmarks tais como HPF, HPF+ [6] e ZPL. A utilização dessas linguagens para o desenvolvimento de novos benchmarks permite a avaliação da eficiência das respectivas linguagens.

Este trabalho é considerado como uma parte de um projeto maior cujo objetivo é desenvolver um modelo para determinar o grau de portabilidade de um benchmark paralelo.

Referências

[1] C. A. Addison, V. S. Getov, A. J. G. Hey, R. W. Hockney, and I. C. Wolton. Benchmarking for Distributed Memory Parallel Systems: Gaining Insight from Numbers. *Parallel Computing*, 1994.

[2] V. Aslot, M. Domeika, R. Eigenmann, G. Gaertner, W. B. Jones, and B. Parady. SPEComp: A New Benchmark Suite for Measuring Parallel Computer Performance. *Lecture Notes in Computer Science*, 2104, 2001.

[3] D. Bailey, E. Barszcz, J. Barton, D. Browning, R. Carter, L. Dagum, R. Fatoohi, S. Fineberg, P. Frederickson, T. Lasinski, R. Schreiber, H. Simon, V. Venkatakrishnan, and S. Weeratunga. The NAS PARALLEL Benchmarks. Technical report, NASA Ames Research Center, 1994. RNR-94-001.

[4] Mohammad Banikazemi, Rama Govindaraju, Robert Blackmore, and Dhabaleswar K. Panda. MPI-LAPI: An Efficient Implementation of MPI for IBM RS/6000 SP systems. *IEEE Transactions on Parallel and Distributed Systems*, 12(10):1081–1093, 2001.

[5] A. Beguelin, A. Geist, J. Dongorra, W. Jiang, and R. Manchek. *PVM: Parallel Virtual Machine. A User Guide and Tutorial for Networked Parallel Computing.* The MIT Press, 1994.

[6] S. Benker and M. Pantano. HPF+ Optimizing HPF for Advanced Applications. *Supercomputer*, 13(2):31–43, Janeiro 1997.

[7] J. Bourgeois and F. Spies. Performance Prediction of an NAS Benchmark Program with ChronosMix Environment. *Lecture Notes in Computer Science*, 1900, 2001.

[8] J. Bourgeois, F. Spies, and M. Trhel. Performance Prediction of Distributed Applications Running on Network of Workstations. In *In Proc. of PDPTA'99*, volume 2, pages 672–678, 1999.

[9] Julien Bourgeois and François Spies. Performance Prediction of an NAS Benchmark Program with ChronosMix Environment. *Lecture Notes in Computer Science*, 1900, 2001.

[10] F. Cappello and D. Etiemble. MPI versus MPI+OpenMP on the IBM SP for the NAS Benchmark. *Super Computing*, 2000.

[11] B. Chamberlain, S. J. Deitz, and L. Snyder. A Comparative Study of the NAS MG Benchmark Across Parallel Languages and Architectures. In *ACM/IEEE Supercomputing Conference on High Performance Networking and Computing*, 2000.

[12] O. A. C. Cortes. Desenvolvimento e Avaliação de Algoritmos Numéricos Paralelos. Master's thesis, Instituto de Ciências Matemáticas e de Computação - USP, São Carlos-SP, 1999.

[13] O. A. C. Cortes and O. R. Saavedra. O Estado da Arte da Lnguagem Fortran. In *X Seminário Regional de Informática*, URO - Santângelo, Novembro 2000. Anais em CD-ROM.

[14] V. K. Decyk, C. D. Norton, and B. K. Szymansky. Object Oriented Programming with Fortran90. http://olympic.jpl.nasa.gov/Reports/Highlights96/decyk.html, 1996.

[15] R. Dimitrov and A. Skjellum. Impact of Latency on Applications Performance. In *Fourth MPI Developer's and User's Conference*, Abril 2000.

[16] G. H. Emilio. *A Methodology for Design of Parallel Benchmarks*. PhD thesis, University of Southampton, Inglaterra, Dezembro 1996.

[17] D. Fliegner, A. Retey, and J. A. M Vermaseren. Parallelizing the Symbolic Manipulation Program Form. *Workshop Science on Cluster Computers*, 2000.

[18] The OpenMP Forum. OpenMP Fortran Application Program Interface - Version 1.0. http://www.openmp.org, 1997.

[19] M. Frumkin, H. Jin, and J. Yan. Implementation of NAS Parallel Benchmark in High Performance Fortran. Technical report, NASA Ames Research Center, 1998. NAS-98-009.

[20] M. Frumkin, M. Schultz, H. Jin, and J. Yan. Implementation of NAS Parallel Benchmark in Java. Technical report, NASA Ames Research Center, 2002. NAS-02-009.

[21] William Gropp and Ewing L. Lusk. Reproducible measurements of MPI performance characteristics. In *PVM/MPI*, pages 11–18, 1999.

[22] D. Grove and P. Coddington. Precise MPI Performance Measurement Using MPIBench. In *HPC Asia*, Setembro 2001.

[23] T. Hey and D. Lancaster. The development of Parkbench and performance prediction. *The International Journal of High Performance Computing Applications*, 14(3):205–215, 2000.

[24] R. W. Hockney. *The Science of Computer Benchmarking*. Society for Industrial and Applied Mathematics, Philadelphia, 1996.

[25] SkaMPI Home Page. Special Karlsruher MPI Benchmarks, 2000. http://linwww.ira.uka.de/ skampi/.

[26] HPFF. *High Fortran Specification*. High Performance Fortran Forum, 1994.

[27] Corp. INTEL. Why Should You Care About Performance? http://pentium.intel.com/procs/perf/intro/bintro.htm, 1998.

[28] H. Jin, M. Frumkin, and J. Yan. The OpenMP implementation of NAS Parallel Benchmark and its Performance. Technical report, NASA Ames Research Center, 1999. NAS-99-011.

[29] K. Kusano, S. Satoh, and M. Sato. Performance Evaluation of the Omni OpenMP Compiler. *Lecture Notes in Computer Science*, 1940, 2000.

[30] C. Lin, L. Snyder, and B. L. Chamberlain. ZPL vs HPF: A Comparison of Performance and Programming Style. Technical report, University of Washington, Washington, 1995. TR 95-11-05.

[31] G. R. Luecke, S. Spanoyannis, and J. Coyle. The Performance of MPI Derived Types on a SGI Origin 2000, a Cray T3E-900, a Myrinet Linux Cluster and an Ethernet Linux Cluster, 2001.

[32] N. MacDonald, E. Minty, T. Harding, and S. Browa. *Writing Message-Passing Parallel Programs with MPI - Course Notes.* The University of Edinburgh, 1994.

[33] B. Mösli. A Comparasion of C++, Fortran 90 e Oberon-2 for Scientific Programming. http://www.arithmetica.ch/Oberon/CFORTRAN/Oberon.html, Maio 1995.

[34] T. Ngo, L. Snyder, and B. Chaberlain. Portable Performance of Data Parallel Languages. In *Procs. of the SC '97: High Performance Networking and Computing*, 1997.

[35] Pallas. Pallas MPI Benchmark - PMB. http://www.pallas.com, 2000.

[36] R. Reussner and G. Hunzelmann. Achieving Performance Portability with SKaMPI for High-Performance MPI Programs. In *Computational Science–ICCS 2001, Proc. of ICCS 2001, International Conference on Computational Science, Part II, Special Session on Tools and Environments for Parallel and Distributed Programming, San Francisco, CA, 2001*, volume 2074 of *Lecture Notes in Computer Science*, pages 841–850, Maio 2001.

[37] R. Reussner, T. J. Träff, and G. Hunzelmann. A Benchmark for MPI Derived Datatypes. *Lecture Notes in Computer Science*, 1908, 2000.

[38] R. H. Reussner. *SkaLib: SKaMPI as Library.* University of Karlsruhe, 1999.

[39] Ralf H. Reussner. Recent Advances in SKaMPI. In E. Krause and W. Jäger, editors, *High Performance Computing in Science and Engineering 2000*, Transactions of the High Performance Computing Center Stuttgart (HLRS), pages 520–530. SPRINGER, 2001.

[40] Ralf H. Reussner. Using SKaMPI for developing high-performance MPI programs with performance portability. *Future Generation Computer Systems*, 2003. to appear.

[41] C. L. Satake. Um Estudo Comparativo de Benchmarks Paralelos. Master's thesis, Instituto de Ciências Matemáticas e de Computação - USP, São Carlos-SP, 1999.

[42] Computer Science Departament. Llcbench home page. http://icl.cs.utk.edu/projects/llcbench/, 1999.

[43] L. Snyder. *A Programmers Guide to ZPL.* Departament of Computer Science and Engineering - University of Washington, 1999.

[44] T. Takahashi, S. Sumimoti, A. Hori, H. Harada, and Y. Ishikawa. PM2: A High Performance Communication Middleware for Heterogeneous Network Environments. In *Proceedings of the on Supercomputing*, 2000.

[45] T. Veldhuizen. C++ Versus Fortran. *Dr. Bobb's Journal - Numeric Programming*, 1997.

[46] R. F. Wijngaart. NAS Parallel Benchmark Version 2.4. Technical report, NASA Ames Research Center, 2002. NAS-02-007.

Web Services: estudo de caso envolvendo uma aplicação B2B

Cristiano Fornari Colpani (FURB)
cristiano.colpani@senior.com.br

Alexander Roberto Valdameri (FURB)
arv@furb.br

Resumo. Este artigo descreve um estudo de caso que apresenta soluções para interconexão de aplicações Business to Business (B2B), onde foram implementados softwares capazes de possibilitar a troca de catálogos de produtos entre aplicações distintas, utilizando uma arquitetura baseada em Web Services desenvolvidos em ambiente Delphi e acessados por aplicações desenvolvidas na plataforma Microsoft .NET. Aplicações baseadas em Web Services permitem a integração com praticamente todas as plataformas de desenvolvimento pois são baseados em padrões especificados pelo World Wide Web Consortiun (W3C) e utilizam padrões para sua implementação, como o protocolo HTTP, SOAP e WSDL e XML.

Palavras-chave: Web Services, SOAP, XML, Microsoft .NET, B2B.

1 Introdução

Com o avanço dos sistemas de informação e a necessidade de se manter todos os processos de negócio integrados, cresce a demanda de tecnologias que possibilitem o desenvolvimento de sistemas com capacidade de integração e interoperabilidade. A integração com outros recursos, sejam eles sistemas de informação ou dispositivos de coleta e geração de dados, tem se mostrado como um grande obstáculo quando as partes envolvidas não possuem mecanismos para receber, processar e compartilhar seus recursos da forma mais transparente possível.

Para Kalakota (2002) o *e-commerce* é a realização de toda a cadeia de valor dos processos de negócio num ambiente eletrônico, por meio da aplicação intensa das tecnologias de comunicação e de informação. As tecnologias que tornam possíveis os processos de *e-commerce* não são apenas a Internet e nas páginas *Web*. É necessária uma retaguarda de aplicações que permitam que a informação esteja disponível e possa ser manipulada através de meios eletrônicos, de forma consistente. Evidentemente este modelo de aplicações interfuncionais pode se tornar bastante complexo e, para que sua implementação torne-se viável, é necessário que tais aplicações sejam baseadas em camadas independentes.

As aplicações de *e-commerce* só tornam-se possíveis se implementarem grande parte de suas funcionalidades integradas as outras aplicações empresarias. Segundo W3C (2002), o protocolo *Simple Object Access Protocol* (SOAP) é designado para troca de informações e processamento remoto entre ambientes descentralizados ou distribuídos. É baseado em linguagem *Extensible Markup Language* (XML) e consiste em três partes: um envelope que define a estrutura para descrever o conteúdo da mensagem e como processá-lo, um conjunto de regras de codificação para especificar instâncias de tipos de dados pertencentes às aplicações e um mecanismo para processamento remoto.

Segundo Marchal (2000), XML é uma linguagem usada para descrever e manipular dados organizados de forma estruturada e uma de suas principais aplicações é a troca de dados entre organizações. Durante anos, isso foi resolvido com tecnologias de *Eletronic Data Interchange*

(EDI) – intercâmbio eletrônico de dados, porém, todos os esforços para sua padronização não chegaram a um consenso, pois as organizações que o utilizavam precisavam de aplicações específicas e que não são fornecidas com recursos de uma forma padronizada.

Como a linguagem XML especifica apenas a estrutura do documento e suas regras de formação, é necessário um protocolo para controlar a transmissão dos dados entre as aplicações. No protocolo SOAP essa transmissão é feita através da associação a um protocolo utilizado na Internet, como o protocolo *Hipertext Transfer Protocol* (HTTP). Para compreender melhor essa associação, Snell (2001) relata em sua obra que essa arquitetura provê uma nova maneira de ver e implementar a integração e interoperabilidade entre aplicações fazendo com que a plataforma de desenvolvimento seja irrelevante. Duas aplicações, sem levar em consideração o sistema operacional, linguagem de programação e qualquer outro detalhe de implementação, comunicam-se utilizando mensagens XML sobre protocolos de Internet como HTTP. O protocolo SOAP é a especificação que detalha como a informação deve ser organizada para prover mecanismos de troca de mensagens e processamento remoto.

A idéia é aplicar os conceitos relativos a *Web Services* na implementação de um protótipo de software para troca de catálogos de preços e informações de produtos, caracterizando uma aplicação *Business to Business* (B2B) que provê recursos para integração com outras aplicações de negócio.

2 Aplicações de Comércio Eletrônico

A integração entre os meios de comércio tradicionais e o *e-commerce* está acontecendo diante dos olhos de todos. A sabedoria convencional diz que o comércio eletrônico é um solvente econômico. Ele dissolve os velhos modelos de negócios, muda a estrutura de custo e formula as conexões entre compradores, vendedores e intermediários. Apesar disto, somente agora está se tornando claro que o comércio eletrônico é também um solvente de relações, dissolvendo fronteiras tradicionais entre clientes e parceiros. Kalakota (2002) separa a evolução do comércio eletrônico em três fases, sendo que a fase atual é descrita como a terceira, que começou em 2000 e é baseada em como a Internet pode influenciar a lucratividade. Lucratividade não significa aumentar somente a receita bruta, mas aumentar as margens de lucro. Essa fase pode ser chamada essencialmente de *e-business*, e inclui todas as aplicações e processos que permitem a uma corporação realizar transações de negócios de forma eletrônica. Além de englobar o comércio eletrônico, o *e-business* inclui atividades de contato e retaguarda que formam o mecanismo principal do negócio moderno. Assim, o *e-business* não trata apenas das transações de comércio eletrônico ou de compras e vendas pela Internet, mas sim com uma estratégia global de redefinição dos antigos modelos de negócios, com o auxílio de tecnologia, para maximizar o valor do cliente e conseqüentemente dos lucros obtidos.

Neste sentido surgiram as aplicações B2B. A sigla B2B provém do termo em inglês *Business to Business* que significa “negócios para negócios”. Abiteboul (2000) explica que as aplicações B2B normalmente integram-se aos sistemas coorporativos das empresas para que possam obter informações sobre os estoques, produção, necessidades de compra de matéria prima e assim disponibilizar recursos eletrônicos para que os fornecedores dessas empresas possam oferecer seus produtos. Uma grande vantagem deste tipo de recurso é que fornecedores podem estabelecer relações comerciais não importando os limites de horário e distância.

Para disponibilizar uma solução de *e-commerce*, seja ela no modelo B2B ou B2C, os sistemas de informação que subsidiam dados e recursos precisam integrar-se de forma automática e segura. Um dos principais desafios no desenvolvimento de soluções integradas envolvendo sistemas heterogêneos é manter o máximo da funcionalidade de cada sistema, obtendo proveito dos seus dados para alimentar outros sistemas, tendo que desenvolver o mínimo de funcionalidade redundante entre eles. Soluções obtidas neste modelo trazem evoluções tecnológicas que permitem que seja possível agregar novas funcionalidades e principalmente aproveitar da melhor forma o legado já existente.

3 Protocolo SOAP

Segundo Seely (2002), *Simple Object Access Protocol* (SOAP) é um mecanismo para interconexão de aplicações formado por três diferentes partes:

- mecanismo de serialização, formado por regras de codificação;
- modelo de empacotamento chamado envelope SOAP;
- mecanismo *Remote Procedure Call* (RPC).

O protocolo SOAP foi especificado pelo *Word Wide Web Consortium* (W3C) que desenvolveu as três partes para serem usadas juntas, porém não impedindo o desenvolvedor de usar cada parte separadamente. Por exemplo, um programa pode utilizar o mecanismo de serialização para gravar um arquivo de configuração ou um sistema de troca de mensagens instantâneas pode utilizar o envelope SOAP para transmitir os dados entre as partes envolvidas. Por outro lado, com o mecanismo RPC pode-se usar envelope SOAP e as regras de codificação para fazer chamadas de função. Essa independência entre as três partes faz com que implementações baseadas em SOAP sejam modulares. Seely (2002) salienta que implementações modulares funcionam bem porque as várias partes desenvolvidas não dependem umas das outras.

3.1 Web Services

O uso de *Web Services* no mundo do software está crescendo rapidamente por possibilitar de forma transparente a interoperabilidade e comunicação entre aplicações. Estes serviços provêem um modelo para implementar a comunicação entre aplicações de diferentes tipos e plataformas.

Segundo W3C (2002), o *Web Service* é um software identificado por uma URL, do qual as interfaces podem ser descritas e descobertas por instrumentos baseados em XML e ainda suportar interações diretas com outros softwares utilizando mensagens em XML transportadas sobre protocolos de Internet.

Scribner (2001) comenta que a maioria das implementações de *Web Services* são construídas sobre o protocolo HTTP, utilizando o protocolo SOAP como padrão para as mensagens e o uso da WSDL como forma para descrever as interfaces.

Todo documento WDSL define um serviço como uma coleção de portas. Essas portas são URLs e definem a localização do serviço. Para que esses serviços possam responder as solicitações, as mesmas precisam estar de acordo com o que o *Web Service* está esperando.

3.2 Utilização de Web Services

Visivelmente uma das áreas mais indicadas para a implementação de *Web Services* é a integração de processos de negócio em aplicações B2B. Por mais de duas décadas os

desenvolvedores vêm tentando integrar processos de negócio utilizando uma combinação de software e protocolos de comunicação.

A capacidade dos *Web Services* em integrar aplicações que implementam processos de negócio é devida à sua interoperabilidade, pois é possível implementar sua interação com praticamente todos os tipos de aplicação.

Os protocolos de para chamada de funções remotas (RPC) mais comuns nunca conseguiram prover uma solução satisfatória para esse problema. Enquanto era possível fazer com que sistemas muito diferentes se comunicassem, o tempo envolvido para fazer isto normalmente não compensava os benefícios. Uma das maiores dificuldades em fazer sistemas diferentes se comunicarem é que os fornecedores escolhem padrões que podem especificamente trazer melhoras na performance de suas próprias plataformas. A interpretação de padrões é outra causa da incompatibilidade entre os protocolos, pois normalmente as especificações são complexas e não podem ser entendidas sem a presença do fornecedor

Segundo Seely (2002), outra funcionalidade importante designada aos *Web Services* é a integração de sistemas heterogêneos, pois a maioria das empresas possui sistemas computacionais distribuídos em diversas plataformas. A integração destes sistemas pode trazer redução nos custos e evolução tecnológica de todo o sistema.

4 Estudo de Caso

Para ilustrar os conceitos associados a *Web Services,* procurou-se desenvolver um protótipo de sistema para troca de catálogos de preços de produtos. Como um *Web Service* é um recurso para acessar e publicar métodos de uma aplicação, tornou-se necessário o desenvolvimento de duas aplicações de comércio eletrônico que fazem acesso ao sistema de troca de catálogos e formam assim o conceito de uma aplicação B2B. Assim sendo, as funcionalidades estarão disponíveis através de métodos remotos que são acessados por outras aplicações.

A aplicação para troca de catálogos fica disponível em um servidor HTTP e suas funcionalidades podem ser acessadas por todas as aplicações que tenham acesso a este servidor. Se este servidor HTTP estiver disponível na Internet então a aplicação poderá ser acessada por qualquer outra aplicação que tenha acesso a Internet e implemente o protocolo SOAP como cliente.

Para possibilitar a interconexão entre todas essas partes é necessário que as mesmas utilizem um padrão como o SOAP para se comunicar. O conjunto da implementação do software para troca de dados com as aplicações de comércio eletrônico implementadas como exemplo de funcionalidade, mostra como é possível fazer a integração entre esses sistemas e que essa integração é possível mesmo que as plataformas de desenvolvimento ou ambiente sejam diferentes, pois cada aplicação é executada em uma camada independente

4.1 Implementações

A seguir são descritas as implementações realizadas para atender o estudo de caso proposto.

4.1.1 Aplicação Principal

Este recurso funciona basicamente como um módulo Web, que é utilizado como base para desenvolvimentos de CGIs comuns. Porém, para que esta aplicação passe de um CGI comum para

um Web Service são adicionados alguns componentes que habilitam as funcionalidades necessárias.

Em um *Web Service* é possível verificar todos os objetos que estão publicados através da solicitação do documento WSDL. Uma aplicação cliente acessa o servidor a partir de regras definidas no WSDL de cada aplicação servidora. A implementação baseada no ambiente Delphi disponibiliza o WSDL adicionando-se o sufixo /wsdl/NomeDaInterface. Existe ainda um recurso interessante que gera um documento HTML com a lista de todos os objetos disponíveis e o respectivo link para seu WSDL. Este documento pode ser obtido adicionando apenas o sufixo /wsdl ao final do caminho do servidor onde se encontra a aplicação. A figura 1 mostra o documento com todas as interfaces pertencentes ao protótipo. Esse documento é gerado fazendo-se uma requisição ao site http://localhost/catalogexchange/ce.exe/wsdl.

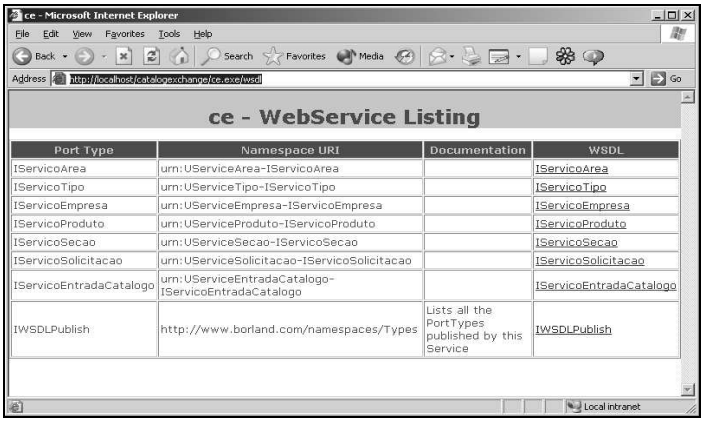


Figura 1: Documento WSDL

4.1.2 Solicitação e Recebimento de Catálogos

Na implementação das telas de solicitação e recebimento de catálogos utilizou-se a ferramenta Visual Studio .Net, porém qualquer outra ferramenta que permita a incorporação de acesso a *Web Services* poderá fazê-lo. O objetivo desta implementação é exemplificar como uma aplicação de comércio eletrônico pode integrar-se à aplicação principal para solicitar e receber catálogos. Para cada objeto disponível na aplicação principal foi criada uma tela que permite o cadastro e a visualização dos dados. Essas telas fazem acesso aos métodos dos objetos disponíveis. O modelo básico pode ser observado na figura 2.

Este modelo de tela é utilizado em toda parametrização do sistema e foi implementado em linguagem C#. O ambiente Visual Studio .Net oferece vários recursos para tornar a implementação de aplicações que acessam *Web Services* fácil e transparente. Após montar o layout básico da página são adicionados objetos para fazer acesso a aplicação principal. Este processo é bastante intuitivo e não requer a codificação manual de grande parte do código fonte necessário para gerar a tela.

Além de montar o layout da tela de definir os objetos de acesso são necessárias algumas linhas de código C# para manipular os dados obtidos no acesso aos objetos da aplicação principal. Essas linhas de código estão normalmente ligadas aos eventos dos botões de manipulação (inserir, atualizar, excluir, refresh) e basicamente tratam de passar os parâmetros para os objetos que fazem

acesso a aplicação principal. Um exemplo deste código é mostrado no quadro 1, onde é tratado o evento de exclusão.

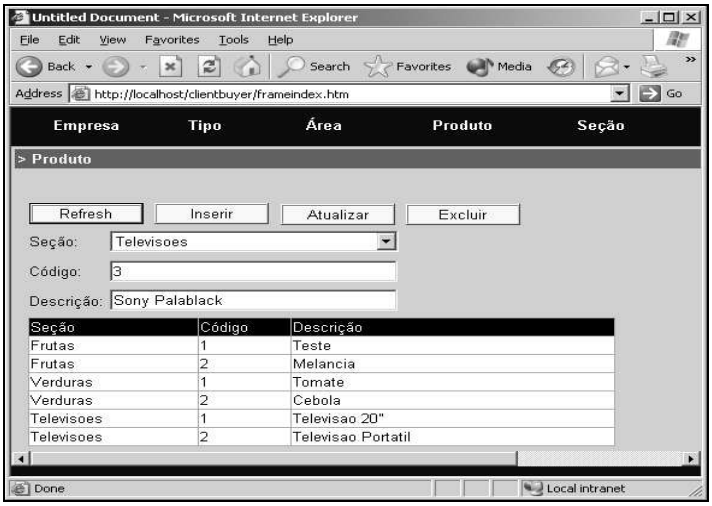


Figura 2: Modelo de tela

```
Private void btExcluir_Click(object sender, System.EventArgs e)
{
  if (tbCodigo.Text.Trim() == "")
  {
    lbError.Visible = true;
    lbError.Text = "Erro: Informe o campo Código";
  }
  else
  {
    int Secao  = Convert.ToInt32(ddArea.SelectedItem.Value);
    int Codigo = Convert.ToInt32(tbCodigo.Text);
    try
    {
      IProduto.DeletaProduto(Secao, Codigo);
    }
    catch(Exception ecp)
    {
      lbError.Visible = true;
      lbError.Text = ecp.Message;
    }
  }
  btRefresh_Click(sender, e);
}
```

Quadro 1: Evento que acessa remotamente o Objeto Iproduto

4.1.3 Envio de Catálogos

O objetivo desta implementação é mostrar que as aplicações desenvolvidas em compatibilidade com o protocolo SOAP podem ser acessadas a partir de diferentes linguagens de programação e de diferentes ambientes.

Esta aplicação simula uma aplicação básica que manipula o estoque de produtos de uma empresa que deseja enviar catálogos relativos a esses produtos a empresas possivelmente compradoras. Para isto esta aplicação interage com a aplicação principal para verificar quais são as empresas que estão cadastradas na aplicação e desejam receber catálogos. Verificadas as solicitações podem ser enviados catálogos para suprir essas solicitações.

A implementação de clientes para aplicações SOAP, no ambiente Borland Delphi 6 inicia como a implementação de uma aplicação normal. A diferença é que nas aplicações típicas a manipulação dos dados é feita utilizando-se diretamente o banco de dados, e em uma aplicação que utiliza objetos localizados em outro servidor, a manipulação de dados é feita através de requisições a objetos remotos utilizando o protocolo SOAP.

4.2 O Processo de Troca de Catálogos

O processo de troca de catálogos utilizando o protótipo desenvolvido pode ser dividido em três partes distintas:

a) parametrização do sistema: onde o sistema recebe as informações sobre as empresas e produtos a serem manipuladores durante o processo;

b) solicitação e recebimento: baseado nas parametrizações anteriores é possível criar solicitações de entrada de catálogo por parte das empresas compradoras. O recebimento é verificado após o envio de catálogos (item c);

c) envio de catálogos: acessando os objetos da aplicação de troca de catálogos é possível verificar que empresas estão solicitando catálogos e proceder o envio.

Para exemplificar a funcionalidade de todas essas partes o sistema receberá o cadastro de empresas dos ramos de eletrodomésticos e som automotivo. A empresa do ramo de eletrodomésticos desejará receber catálogos de aparelhos de ar-condicionado e máquinas de lavar roupa. A empresa do ramo de som automotivo desejará receber catálogos de aparelhos de *cd player* do tipo simples ou *changer*.

A parametrização do sistema ocorre quando existe a necessidade do sistema ser preparado para um processo de troca de catálogos. Esta parametrização é feita através do acesso via *browser* à aplicação de parametrização, que foi implementada segundo os conceitos de uma página *Web*, fazendo acesso aos objetos da aplicação principal. A tela inicial da aplicação de parametrização pode ser observada na figura 4, onde é exibido um menu contendo os cadastros a serem parametrizados (empresa, tipo, área, produto, seção) e a primeira página de cadastro relativa ao cadastro de empresas.

Ao cadastrar uma empresa deve-se informar seu tipo (vendedora ou compradora), a área de atuação, informação esta importante para que as empresas vendedoras possam enviar os catálogos, um código, que é o identificador do registro e uma descrição para identificar a empresa.

A interface permite customizar as empresas envolvidas no processo de troca, bem como as seções, áreas, tipos e produtos do catálogo que serão solicitados.

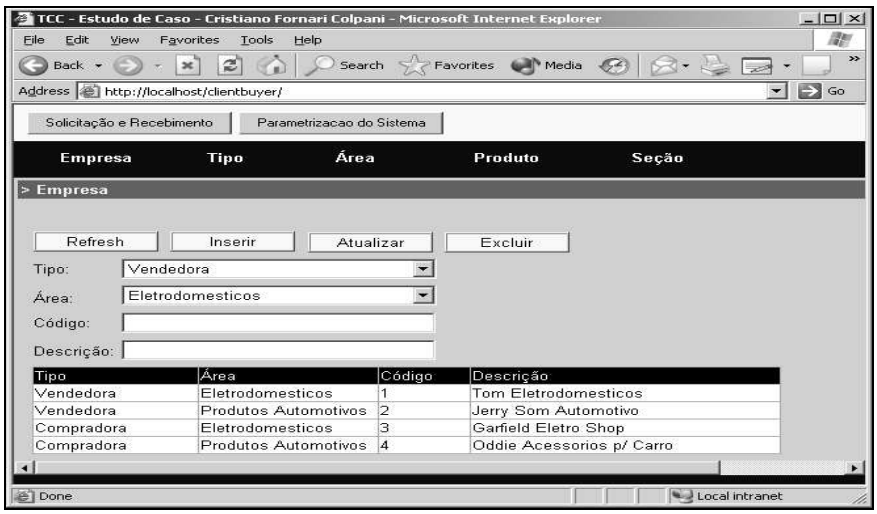

TCC - Estudo de Caso - Cristiano Fornari Colpani - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/clientbuyer/
Go
Solicitação e Recebimento
Parametrizacao do Sistema
Empresa Tipo Área Produto Seção
> Empresa
Refresh
Inserir
Atualizar
Excluir
Tipo: Vendedora
Área: Eletrodomesticos
Código:
Descrição:
Tipo | Área | Código | Descrição
Vendedora | Eletrodomesticos | 1 | Tom Eletrodomesticos
Vendedora | Produtos Automotivos | 2 | Jerry Som Automotivo
Compradora | Eletrodomesticos | 3 | Garfield Eletro Shop
Compradora | Produtos Automotivos | 4 | Oddie Acessorios p/ Carro
Done
Local intranet

Figura 4: Parametrização

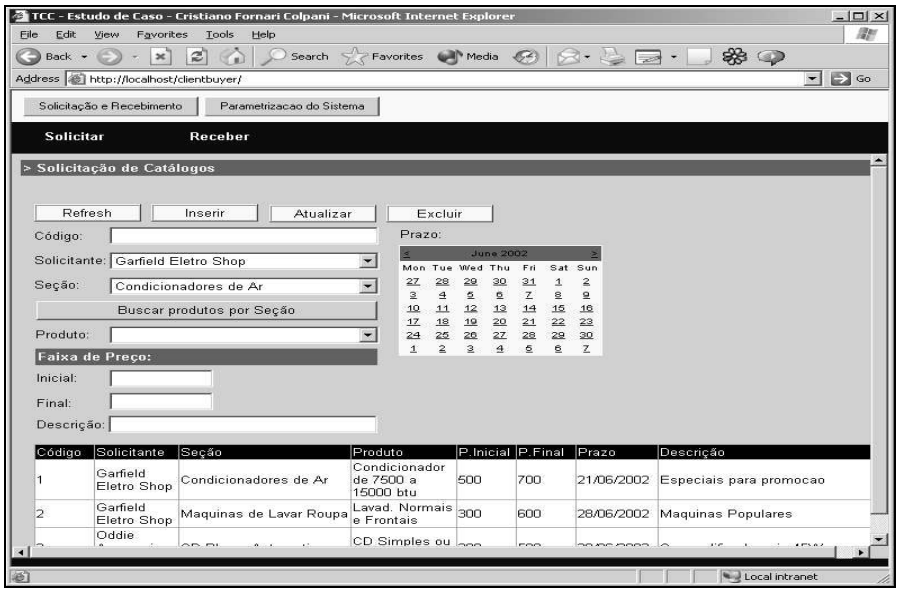

TCC - Estudo de Caso - Cristiano Fornari Colpani - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/clientbuyer/
Go
Solicitação e Recebimento
Parametrizacao do Sistema
Solicitar Receber
> Solicitação de Catálogos
Refresh
Inserir
Atualizar
Excluir
Código:
Prazo:
Solicitante: Garfield Eletro Shop
Seção: Condicionadores de Ar
Buscar produtos por Seção
Produto:
Faixa de Preço:
Inicial:
Final:
Descrição:
June 2002
Mon Tue Wed Thu Fri Sat Sun
Código | Solicitante | Seção | Produto | P.Inicial | P.Final | Prazo | Descrição
1 | Garfield Eletro Shop | Condicionadores de Ar | Condicionador de 7500 a 15000 btu | 500 | 700 | 21/06/2002 | Especiais para promocao
2 | Garfield Eletro Shop | Maquinas de Lavar Roupa | Lavad. Normais e Frontais | 300 | 600 | 28/06/2002 | Maquinas Populares
Local intranet

Figura 5: Solicitação de Catálogo

A solicitação e recebimento de catálogos ocorrem quando uma das empresas compradoras deseja solicitar catálogos para algum produto e posteriormente verifica o recebimento. Tais solicitações também são cadastradas através de uma interface HTML acessada via *browser*. Esta tela pode ser observada na figura 5 e pode ser acessada a partir do menu "Solicitar".

Pode-se observar na figura 5 o cadastramento de 3 solicitações. Duas feitas pela empresa Garfield Eletro Shop e uma feita pela empresa Oddie Som Automotivo. Essas solicitações serão visualizadas pela aplicação que faz o envio de catálogos que é discutida na seção a seguir. Após

execução do processo de envio dos catálogos por parte das empresas vendedoras, o cliente recebe as informações relativas aos catálogos.

A figura 6 mostra os catálogos que foram recebidos para as solicitações anteriormente cadastradas.

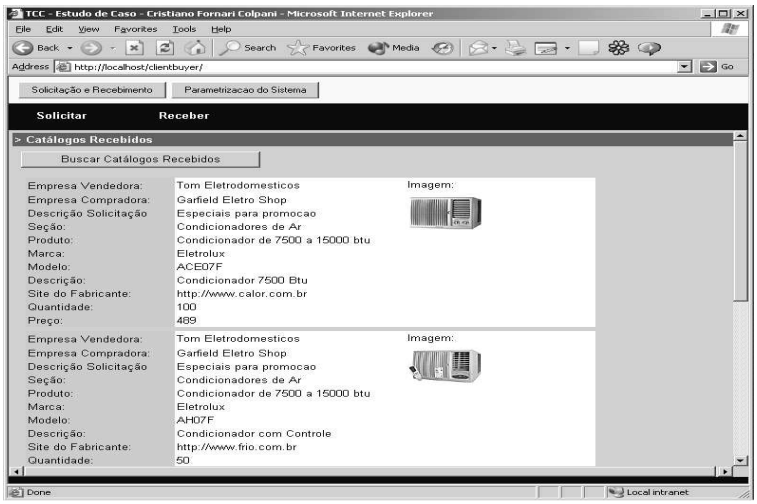


Figura 6: Catálogos Recebidos

O envio de catálogos ocorre quando é verificada a necessidade de envio de informações por alguma empresa vendedora. A figura 7 mostra a aplicação de envio de catálogos pronta para enviar catálogos. A interface da aplicação de envio de catálogos consiste na seleção da área da empresa que será responsável pela venda seguida do nome da empresa.

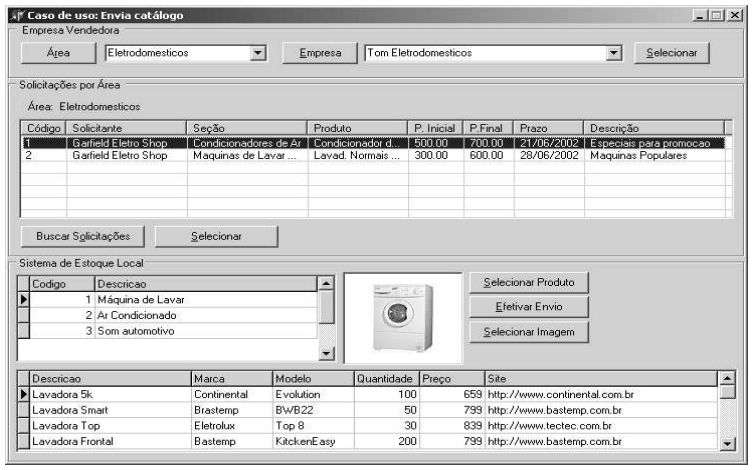

Caso de uso: Envia catálogo
Empresa Vendedora
Área
Eletrodomesticos
Empresa
Tom Eletrodomesticos
Selecionar
Solicitações por Área
Área: Eletrodomesticos
Código | Solicitante | Seção | Produto | P. Inicial | P.Final | Prazo | Descrição
1 | Garfield Eletro Shop | Condicionadores de Ar | Condicionador d... | 500.00 | 700.00 | 21/08/2002 | Especiais para promocao
2 | Garfield Eletro Shop | Maquinas de Lavar ... | Lavad. Normais ... | 300.00 | 600.00 | 28/06/2002 | Maquinas Populares
Buscar Solicitações
Selecionar
Sistema de Estoque Local
Codigo | Descricao
1 | Máquina de Lavar
2 | Ar Condicionado
3 | Som automotivo
Selecionar Produto
Efetivar Envio
Selecionar Imagem
Descricao | Marca | Modelo | Quantidade | Preço | Site
Lavadora 5k | Continental | Evolution | 100 | 659 | http://www.continental.com.br
Lavadora Smart | Brastemp | BWB22 | 50 | 799 | http://www.bastemp.com.br
Lavadora Top | Eletrolux | Top 8 | 30 | 839 | http://www.tectec.com.br
Lavadora Frontal | Bastemp | KitckenEasy | 200 | 799 | http://www.bastemp.com.br

Figura 7: Aplicação para Envio de Catálogos

5 Considerações Finais

O protótipo desenvolvido mostrou a viabilidade da utilização de *Web Services* na implementação de aplicações B2B. Esta constatação se deu principalmente pela flexibilidade na distribuição dos dados através do desenvolvimento de aplicações em diferentes plataformas e linguagens de programação, interligadas através dos objetos baseados em SOAP.

O protocolo SOAP apresenta uma séria de vantagens, tais como a diversidade de chamadas de funções remotamente como DCOM ou CORBA, é simples de implementar, testar e usar, é um padrão criado por um consórcio entre várias empresas, que utiliza em boa parte os padrões da Web: a comunicação é feita via HTTP com dados formatados em documentos XML.

Unindo os conceitos das especificações HTTP, SOAP, XML juntamente com o WSDL foi possível criar o conceito de *Web Services*. O uso de *Web Services* no desenvolvimento de sistemas está crescendo rapidamente por possibilitar a interoperabilidade e comunicação entre aplicações. Estes serviços provêem um modelo para empacotar e disponibilizar funções remotas de maneira distribuída, possibilitando assim comunicação entre aplicações de diferentes tipos e plataformas.

6 Referências Bibliográficas

ABITEBOUL, Serge; BUNEMAN, Peter; SUCIU, Dan. **Gerenciando dados na WEB**. Rio de Janeiro: Campus, 2000.

KALAKOTA, Ravi; ROBINSION, Marcia. **E-business**: estratégias para alcançar o sucesso no mundo digital. Porto Alegre: Bookman, 2002.

MARCHAL, Benoit. **XML conceitos e aplicações**. São Paulo: Berkeley, 2000.

SCRIBNER, Kenn; STIVER, Mark C. **Applied SOAP**: implementing .NET XML web services. Indianapolis: Sams, 2001.

SNELL, James; TIDWELL, Doug; KULCHENKO, Kulchenko. **Programming web services with soap**. Santa Clara: O'Reilly, 2001.

SEELY, Scott. **SOAP**: Cross plataform web service development using XML. Upper Saddle River: Prentice Hall PTR, 2002.

W3C WORLD WIDE WEB CONSORTIUM. **Simple object access protocol (SOAP)**. Estados Unidos, maio 2000b. Disponível em <http://www.w3.org/XML>. Acesso em: 20 fev. 2002.

Utilização da Pesquisa Tabu na geração de um Sistema de Informação Geográfica aplicado ao problema de localização de torres de rádio transmisão

Leandro Toss Hoffmann (UNISINOS)
lth@ig.com.br

Arthur Tórgo Gómez (UNISINOS)
breno@exatas.unisinos.br

Resumo: Este trabalho apresenta uma abordagem para o problema de localização de antenas, comumente encontrado nos projetos de comunicação que utilizam sistemas de rádio. O objetivo deste trabalho consiste em utilizar um protótipo de um Sistema de Informação Geográfica para auxiliar no posicionamento de torres de radiotransmissão, formulado como um Problema de Localização com Máxima Cobertura. Para abordagem do problema é utilizado Pesquisa Tabu e a Heurística Localização-alocação para comparação com a primeira. Para tanto, inicialmente são apresentados conceitos básicos pertinentes à área de geoprocessamento, Sistemas de Informação Geográfica e antenas, necessários para abordagem do problema. Posteriormente, são realizados experimentos com o protótipo de SIG, seguido dos resultados obtidos.

Palavras-chave: Problema de Localização com Máxima Cobertura, Sistemas de Informação Geográfica, Antenas.

1 Introdução

Um problema comumente encontrado nos dias atuais, relativo a análise espacial, é o posicionamento de antenas. O problema da definição do posicionamento de antenas surge da necessidade de se criar um sistema de comunicação utilizando sinais de ondas de rádio. Sendo a tomada de decisão, que define o posicionamento das antenas de transmissão / recepção, realizada baseada nos pontos onde o sinal deverá atingir. Esse problema ocorre, por exemplo, quando é necessário definir onde instalar antenas de um provedor de Internet via rádio, procurando maximizar o número de clientes atendidos com um número mínimo de antenas. Outro exemplo comum, atualmente, é a instalação de torres com antenas para atender a demanda de serviços de telefonia móvel. Na maioria dos casos, o problema passa a ser complexo devido à extensão territorial em questão, onde a capacidade de ganho de uma antena é limitada. Além disto, a tomada de decisão envolve variáveis relativas à propagação do sinal da antena e eventuais problemas de sombra de transmissão, normalmente complexos.

Neste trabalho, busca-se abordar o problema de posicionamento de antenas numa dada região, utilizando-se técnicas de análise de espacial associadas ao Problema de Localização, suportadas por um Sistema de Informação Geográfica, que reúna as tecnologias existentes necessárias para solução do problema.

Desta forma, a primeira parte do trabalho, item 2, descreve-se os conceitos básicos de geoprocessamento, Sistemas de Informação Geográfica e antenas, conceitos estes necessários para a abordagem do problema de localização acima descrito. O item 3, concentra-se na definição do problema proposto, sua formulação matemática e descrição do protótipo. Em seguida, no item 4, é apresentada uma aplicação do protótipo, por meio de algumas estratégias de implementação adotadas. Após, no item 5, experimentos são apresentados, seguido dos resultados no item 6 e a conclusão do trabalho no item 7.

2 Conceitos básicos

Neste item são descritos os conceitos fundamentais necessários para o desenvolvimento do trabalho. Inicia-se abordando aspectos de geoprocessamento, seguido da apresentação dos Sistemas de Informação Geográfica e por último é descrito o funcionamento básico das antenas, bem como suas propriedades eletromagnéticas.

2.1 Geoprocessamento

Baseado em conceitos de cartografia, diversas técnicas foram desenvolvidas para coleta, tratamento, manipulação, análise e apresentação de dados espaciais, ou seja, informações pertinentes a mapas. Atualmente, todas essas tecnologias podem ser resumidas pelo termo geoprocessamento, como pode ser visto em Carvalho (2000). No processo de modelagem de um Sistema de Informação Geográfica, o entendimento das tecnologias pertencentes ao geoprocessamento faz-se necessário, sendo a principal a análise de dados espaciais.

2.1.1 Análise de dados espaciais

Enquanto muitos usuários se limitam a visualizar dados geográficos e tirar conclusões intuitivas, cada vez mais busca-se desenvolver ferramentas computacionais para análise automática desses , citam Câmara *et al.* (2001). Exemplos de aplicações resultantes da análise de dados pode ser a escolha de um local num mapa para instalação de depósito de lixo químico, como em Carvalho *et al.* (2000), ou ainda, tomar medidas para reduzir a mortalidade infantil, baseado na distribuição espacial da ocorrência de casos numa dada região, como em Câmara *et al.* (2001). Quando a análise espacial apresenta problemas que podem ser modelados em redes, pode-se aplicar técnicas de análise de redes para tentar solucionar o problema, segundo Lorena (2001). É possível classificar os problemas modelados em redes como de localização e os relacionados a transportes. Os problemas de localização consistem basicamente na decisão de onde posicionar facilidades, considerando clientes que devem ser servidos, otimizando um certo critério e respeitando restrições do meio. Já os problemas relacionados a transporte consistem em definir rotas entre fonte(s) e destino(s) segundo critérios pré-definidos.

2.2 Sistemas de Informação Geográfica

Um Sistema de Informação Geográfica (SIG) é um sistema computadorizado especializado que permite realizar as mais variadas tarefas referentes ao gerenciamento de dados geográficos. Burrough & McDonnell (1998), Câmara *et al.* (1996), Carvalho *et al.* (2000), Davis & Câmara (2001), Eastman (1998), Maguire *et al.* (1991), Murai (1999), Rossiter (1994) concordam que o SIG é um conjunto de *hardware* e *software* utilizado para armazenar, manipular, analisar e visualizar dados espaciais georreferenciados. Câmara (1995), Câmara *et al.* (1996) e Câmara *et al.* (2000) apresentam a estrutura geral de um SIG através dos seguintes componentes: Interface com o usuário, entrada e integração de dados, funções de processamento gráfico de imagem, visualização e plotagem, e armazenamento e recuperação de dados. Estes componentes se relacionam de forma hierárquica estando o nível mais próximo ao usuário, a interface e no mais interno do sistema o sistema gerenciador de banco de dados geográfico. Os demais componentes encontram-se num mesmo nível intermediário.

2.2.1 Modelo de dados geográficos

Um Sistema de Informação Geográfica trabalha basicamente com dois tipos de informações: dados espaciais e alfanuméricos, como pode ser visto em Eastman (1998). A maior preocupação consiste na forma de como representar os dados espaciais, para tanto, utiliza-se dois tipos principais de modelos de dados: modelo matricial e modelo vetorial, Câmara *et al.* (1996), Câmara (1995), Carvalho *et al.* (2000), Eastman (1998), Murai (1999) e Rossiter (1994). A utilização de

cada modelo de dados varia basicamente em relação ao tipo de dado ou as aplicações. O modelo matricial, também conhecido como formato *raster*, utiliza células de uma matriz dividida regularmente. A localização de objetos geográficos é definida pela posição nas linhas e colunas dessa matriz, segundo Carvalho *et al.* (2000) e Murai (1999). Cada célula terá um valor correspondente ao tema mais freqüente naquela localização espacial, registrando assim, a condição ou atributo da superfície terrestre daquele ponto, explicam Câmara (1995) e Eastman (1998). Já o Modelo Vetorial, representa informações com as coordenadas exatas de sua localização, segundo Rossiter (1994). Para tanto, as feições geográficas (rios, estradas, terrenos) são codificadas em um sistema de coordenadas cartesianas (x,y) utilizando componentes básicos: pontos, linhas e polígnos, Carvalho *et al.* (2000) e Eastman (1998).

2.2.2 Classes de dados geográficos

Para melhor entendimento, as entidades do mundo real podem ser classificadas no SIG em classes de dados geográficos, segundo Câmara *et al.* (1996) e Murai (1999). Câmara *et al.* (1996) e Câmara (1995) classificam dos dados geográficos em: mapas temáticos; mapas cadastrais; redes; modelos numéricos de terreno e imagens. A classe de Modelo Numérico de Terreno, utilizada neste trabalho, representam grandezas que variam continuamente no espaço, Murai (1999) e Rossiter (1994). Em outras palavras, é uma representação matemática da distribuição espacial de uma determinada característica vinculada a uma superfície real, INPE (2001). Sua representação pode ser efetuada de diversas maneiras: grande regular de pontos, chamado de Modelo Digital de Elevação (MDE); grande irregular de pontos, criando Redes Triangulares Irregulares (RTI); linhas de contorno e perfil de pontos, descritos em Murai (1999) e Rossiter (1994).

2.3 Antenas

Uma antena pode ser considerada simplesmente como um tipo especial de linha de transmissão, a qual irradia ou capta energia, segundo Lytel (1973). Martinha (1987), define antenas como “circuitos elétricos capazes de enviar para o espaço a energia que se lhe aplica, ou, inversamente, capazes de captar energia das ondas radioelétricas do espaço”. Através da freqüência de uma onda é possível classificar dois tipos básicos de ondas, segundo sua propagação: ondas terrestres e ondas espaciais. As ondas terrestres se constituem dos sinais diretos entre o transmissor e o receptor, e de sinais provenientes de ondas refletidas pela terra. Já as ondas espaciais referem-se às ondas terrestres propagadas em direção ao espaço, mas refletidas pela ionosfera ou troposfera de volta a terra, como visto em em Lytel (1973). Vassalo (1979) classifica as ondas em quatro formas diferentes de programação: direta, por reflexão, por difração e por refração. Informação adicional sobre a modelagem matemática desses tipos de propagação de ondas podem ser vistas em Smit (1986).

2.3.1 Polarização

As ondas de rádio constituem um forma de radiação eletromagnética, assim, são formadas por um campo elétrico e um campo magnético, os quais são representados por duas senóides, cada campo a 90^o em relação ao outro, como explicado por Lytel (1973). É a direção do campo elétrico de uma onda que determina sua polarização, explica Vassalo (1979). Uma antena transmissora vertical irradia uma onda verticalmente polarizada.

2.3.2 O funcionamento básico das antenas

O dipolo padrão, ou dipolo de meia onda, é o elemento fundamental de um sistema de antenas. Eletricamente, a antena dipolo é uma linha de transmissão de um quarto de comprimento de onda, em circuito aberto, alimentada por um gerador. A antena dipolo de meia onda, também conhecida por dipolo de Hertz, é constituída pela abertura das extremidades dos fios da linha de

transmissão, tendo seu comprimento físico total igual a meio comprimento de onda. A antena dipolo de meia onda é a mais simples das antenas, sendo tomada como referência comparativa para as demais antenas desenvolvidas.

2.3.3 Diagramas de irradiação

Uma antena nunca irradia energia em todas as direções, sendo assim, seu diagrama nunca será uma esfera. O diagrama de irradiação de uma antena dipolo de meia onda, por exemplo, tem um formato toroidal, como visto em Martinha (1987). Assim, estando uma antena verticalmente posicionada ao centro do toróide, sua irradiação se dá em intensidades iguais nas direções norte, sul, leste, oeste, porém de forma desigual nas posições acima e abaixo da antena.

2.3.4 Directividade

A sensibilidade de uma antena em captar ou irradiar sinais numa dada direção é chamada de directividade da antena, a qual é determinada pelo diagrama de irradiação da mesma. A directividade é indicada pelo comprimento de uma reta do centro da antena até a linha do diagrama de irradiação, onde a orientação é definida pela visada da antena até o ponto que se deseja calcular a directividade. A directividade de uma antena dipolo de meia onda é bidirecional, formado por dois lóbulos, podendo ser aproximada pela Equação (1).

$$D = \cos^2 \theta \tag{1}$$

onde, θ é o ângulo de visada formado entre a antena e o receptor

2.3.5 Ganho

O ganho (G) é outra propriedade importante de uma antena, relativa a sua transmissão ou recepção. É uma propriedade comparativa da potência de uma antena em relação à antena dipolo padrão nas mesmas condições e freqüência. Usualmente, a razão entre as potências é expressa em decibéis, segundo Lytel (1973) e Vassalo (1979). Neste trabalho, o ganho de um receptor padrão será expresso pela Equação (2), que traduz uma relação inversa do quadrado da distância entre o receptor e a antena transmissora. Esta relação é ainda proporcional a directividade da antena.

$$G = \frac{1}{d^2} \cos^2 \theta \tag{2}$$

onde: d é a distância euclidiana entre o receptor padrão e a antena transmissora.

3 Desenvolvimento do Protótipo

Dentro da proposta de desenvolver uma ferramenta computacional para suporte à tomada de decisão no posicionamento de antenas, neste item será apresentado o desenvolvimento do protótipo de um Sistema de Informação Geográfica. Assim, o problema é definido e em seguida formulado matematicamente. Após, são apresentadas as técnicas utilizadas para sua abordagem, bem como o protótipo do SIG.

3.1 Definição do problema

Nos projetos de comunicação utilizando sistemas de rádio, uma das preocupações é definir o posicionamento de um dado número de antenas. Baseado na localização dos pontos receptores potenciais, o usuário deve tomar decisões a respeito do tipo de antena a ser utilizado, sua quantidade e a melhor posição viável das torres, de modo a garantir um bom funcionamento do sistema e a legibilidade do sinal. Para tanto, limites de distância de transmissão e a dinâmica do terreno são fatores que devem ser considerados.

O objetivo desse trabalho consiste em utilizar um protótipo de um Sistema de Informação Geográfica que auxilie no posicionamento de torres de radiotransmissão, considerando os seguintes fatores: as coordenadas de pontos para recepção do sinal; a quantidade de clientes por pontos de recepção; as coordenadas das antenas; o modelo de irradiação das antenas transmissoras; a orientação e polarização das antenas transmissoras; a altura das torres das antenas; o modelo em três dimensões do terreno, com quotas de elevação; as regiões viáveis ou proibidas para posicionamento das antenas; e a distância comparativa do receptor ideal. Não está no escopo do protótipo, analisar os modelos de irradiação das antenas receptoras, nem o comportamento das ondas de rádio de reflexão terrestre ou espacial.

3.2 Formulação

Considerando um terreno discretizado, obtido através de uma modelagem matricial, o problema de posicionar antenas pode ser modelado por redes e abordado como um problema de localização de facilidades. Os problemas de localização são classificados conforme seu objetivo, como por exemplo: problemas de cobertura e problemas de localização de medianas, descritos em Lorena (2001). No problema de localização com cobertura o objetivo da análise pode buscar uma solução que: a) maximize um conjunto de clientes atendidos, a partir da localização de um dado número fixo de servidores, caracterizado como Problema de Localização com Máxima Cobertura (*Maximal Set Covering Problem)*; ou b) minimize o número total de servidores localizados num plano, necessários para atender todos clientes, caracterizado como Problema de Cobertura de Conjuntos (*Set Covering Problem*), segundo Goldbarg & Luna (2000), Lorena (2001) e Scaparra & Scutellà (2001). Exemplos com essas aplicações podem ser encontrados em Galvão & Revelle (1996), Galvão *et al.* (2000) e Lorena *et al.* (2001). Já o problema de localização de medianas é um problema clássico abordado pela literatura, onde o objetivo é localizar um número fixo de servidores (ou recursos) de forma a minimizar a soma das distâncias de cada vértice à sua facilidade, como pode ser visto em Lorena (2001).

Neste trabalho, o problema apresentado, consiste em definir o posicionamento de um número fixo de antenas transmissoras, de modo a atender um número máximo de clientes receptores. Sendo assim, o problema será modelado como um Problema de Localização com Máxima Cobertura, onde dado um número fixo de antenas, o objetivo é maximizar o número de clientes receptores, conforme mostrado nas Equações (3), (4), (5), (6) e (7).

A formulação acima caracteriza um problema de Programação Inteira (PI), onde suas variáveis são limitadas a um conjunto fixo de valores, segundo Papadimitriou & Steiglitz (1998) e Rardin (1998). Entre as abordagens tradicionais de PI pode-se destacar *Branch-and-Bound*, *Cutting-plane* e Relaxação Lagrangeana, descritas por Rardin (1998). Todavia, segundo Goldbarg & Luna (2000), “aplicações do mundo real, modeladas por PI, normalmente implicam numa maior complexidade computacional do que as oriundas de situações de não-linearidade de funções”. Na sua abordagem, acabam também sendo utilizadas técnicas heurísticas.

Neste trabalho, foram utilizadas duas técnicas heurísticas para abordagem do problema: Heurística de Localização-alocação e Pesquisa Tabu, esta úlima descrita por Glover (1989). A primeira é uma técnica de busca local utilizada no trabalho com problemas de localização de Lorena (2001), inspirada no trabalho de Cooper (1963), citado por Lorena (2001), e Taillard (1996). Já a Pesquisa Tabu é uma técnica genérica de busca no espaço, utilizada em vários tipos de aplicações, detalhado em Glover & Laguna (2001). Objetiva-se com o uso da Pesquisa Tabu, realizar um estudo comparativo em relação à Heurística de Localização-alocação.

Abordagens heurísticas não garantem a obtenção da solução ótima para um problema, por outro lado, podem representar uma implementação com menor demanda computacional em relação as abordagens tradicionais, como cita Pidd (1998).

$$Max \quad f = \sum_{i=1}^{m} c_i y_i \tag{3}$$

Sujeito a:

$$\sum_{j \in N_i} x_j \geq y_i, i = 1,...,m \tag{4}$$

$$\sum_{j=1}^{n} x_j = p \tag{5}$$

$$x_j \in \{0,1\},\ j = 1,...,n \tag{6}$$

$$y_i \in \{0,1\}, i = 1,...,m \tag{7}$$

Onde: $N_i = \{ j \mid d_{ij} \leq d \}$ é o conjunto de facilidades que atendem um ponto de receptor *i.*

y_i = receptor i

c_i = peso do receptor i

m = conjunto de pontos de receptores

p = número total de antenas admitido

x_j = ponto viável j para localização de antena

dij = distâncias entre os pontos

n = conjunto de pontos viáveis para localização de antenas

3.2.1 Heurística de Localização-alocação

É uma técnica de busca local que visa otimizar soluções de problemas de agrupamentos (*clustering),* Lorena (2001). Através de uma estrutura de grafo, uma solução inicial identifica *p* agrupamentos C^k, $k \in$ {1, 2,..., p}, onde *p* é um número de medianas definido. A heurística de Localização-alocação procura então melhorar a solução inicial, reposicionando as medianas e redefinindo os agrupamentos. O processo é iterativo até que não haja melhorias na função objetivo. Lorena (2001) sugere que o processo de troca entre vértices mediana e não-mediana em cada agrupamento C^k, k = 1,...,*p*, pode ser executada para:

a) Todos os vértices não medianas do agrupamento C^k, ou
b) Apenas para os vértices não-medianas alocados do agrupamento C^k, ou
c) Apenas para os vértices não-medianas localizados a uma certa distância do vértice mediana do agrupamento C^k.

3.2.2 Pesquisa Tabu

É uma técnica desenvolvida por Glover (1989) aplicada a resolução de problemas de otimização combinatorial. Pesquisa Tabu é uma meta-heurística, ou seja, uma estratégia mestre que guia e modifica outras heurísticas para produzir soluções além daquelas que são normalmente geradas por buscas locais, segundo Glover & Laguna (2001) e Hertz (1991).

Através do algoritmo de Pesquisa Tabu, pode-se encontrar em um conjunto *X* de possíveis soluções *s* que otimiza a função objetivo *f*. Essa otimização pode ser tanto maximizar a função *f* quanto minimizá-la, como mostrado a seguir:

Minimizar (Max) $f(s)$: $s \in X$ e $X \in R_v$, sendo R_v é a região viável.

A vizinhança *N(s)* é definida como cada solução *s* de *X*. O algoritmo inicialmente começa com uma solução inicial s_{inic}; sempre que uma possível solução *s* é encontrada, gera-se uma

vizinhança *V** a partir dessa, sendo que a melhor solução de *V** é *s**. Para evitar fazer ciclos em torno de um ótimo local, não é permitido que se admitam *k* soluções já visitadas, onde *k* é um dado número. Para isso, implementa-se uma Lista Tabu *T* de tamanho |*T* | = *k*, que é usada como uma fila circular. Sempre que um movimento *s* para *s** é executado, seu movimento inverso é armazenado em *T*. O movimento armazenado é considerado proibido, ou seja, é um movimento tabu. A lista tabu pode também ser usada para fazer o movimento inverso, partindo do último movimento armazenado até o primeiro.

Existe porém a possibilidade de um movimento tabu causar uma melhora na função *f*. Para que esse movimento tabu possa ser admitido, tem-se uma função critério de aspiração *A(z).* Se o movimento para uma solução *si* for um movimento tabu, dado que *f(si)* <= *A(z=F(s))* então o movimento é admitido, passando a fazer parte de *V**. O critério de parada da pesquisa é quando o algoritmo atinge um número de iterações previstas (parâmetro NBMAX) ou quando o programa não gera vizinhança devido a lista tabu ser muito grande ou a vizinhança ser muito pequena.

3.3 Descrição do Protótipo

Nessa sessão é descrito o modelo do protótipo do Sistema de Informação Geográfica que auxiliará o usuário na tomada de decisão de posicionamento de antenas. O modelo é uma representação simplificada do comportamento do protótipo e descrição de suas entradas e saídas. Em seguida, a modelagem das entidades externas ao sistema é exposta.

3.3.1 Modelo do Protótipo

O modelo do protótipo, representado pela Figura 1, descreve o processo de otimização do posicionamento de antenas, desde a estrada dos dados até a saída do sistema. Observa-se que os dados de entrada, constituídos pelo modelo das entidades terreno, antenas e receptores, são validados e integrados. Em seguida, caso as antenas não estejam pré-posicionadas, é feita a geração das posições inicias. Antes de iniciar a otimização, é delimitado o espaço de busca de pontos viável para posicionamento das antenas no terreno. Este processo visa reduzir a quantidade de pontos viáveis no terreno, agilizando o processo de otimização, e será descrito posteriormente no item 4.1. O processo de otimização do posicionamento das antenas é realizado após a escolha da heurística, e antes que os resultados sejam informados ao usuário, são gerados dados adicionais para futura análise visual em três dimensões (*3D*) da posição das antenas e dos receptores no terreno. A saída do modelo consiste na localização das antenas determinada pelo processo de otimização, além da relação dos receptores atendidos.

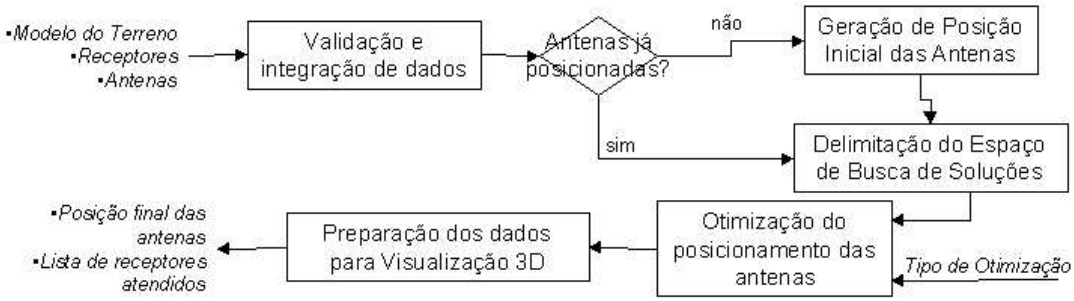


Figura 1 – Modelo do protótipo de SIG.

3.3.2 Modelagem das entidades externas

Os dados de entrada do sistema constituem as entidades externas ao sistema e são representadas na forma de instâncias de objetos no sistema. Essas entidades podem ser constituídas

por dados espaciais ou alfanuméricos e as estratégias adotadas para sua modelagem são descritas a seguir.

Terreno: o terreno representa a superfície na qual os receptores estão localizados e as antenas serão posicionadas. A sessão 2.2.1 descreveu dois tipos básicos de modelagem de terreno: vetorial e matricial. Em se tratando de uma aplicação que necessite efetuar análises de dados espaciais em três dimensões, é comum utilizar-se o MNT em formato matricial. Desta forma, as quotas de elevação do terreno, em coordenadas discretas, serão armazenadas em células de uma matriz *n* x *m*. O modelo desenvolvido para o protótipo pode ainda armazenar informações sobre a classificação do terreno, na forma de uma camada adicional do mapa. Assim é possível identificar áreas proibidas para posicionamento de antenas, tais como, água, estradas, matas, entre outros.

Antena: além das propriedades de propagação, descritas no item 2.3.3, a antena contará com atributos que descrevem sua fixação numa dada coordenada. Esses atributos são: altura da torre, polarização (horizontal / vertical) e o sentido da antena (Norte, Sul, Leste, Oeste).

Receptor: um receptor é toda e qualquer entidade, com posição fixa, que se deseja cobrir com o sinal da antena. Além de informações sobre sua localização no mapa matricial, a modelagem do objeto permite: estipular-lhe uma altura extra para o receptor, em relação ao solo; atribuir-lhe um peso, privilegiando assim o receptor ou identificando *n* receptores para uma mesma coordenada do mapa; e identificar a antena que o atende, com o respectivo ganho.

4 Aplicação

Este item descreve as etapas de implementação do protótipo proposto. O objetivo não é descrever exaustivamente o processo de implementação, mas apresentar as técnicas mais importantes escolhidas para compor o software. Para implementação do modelo foi utilizada a linguagem C++, por se tratar de uma linguagem portável, com boa performance de execução, quando comparada a outras linguagens que utilizam sistemas de coleta de lixo, como citam Lee & Tepfenhart (2001).

4.1 Delimitação do espaço de busca

Dado que o terreno foi modelado através de uma estrutura matricial, a localização das antenas e dos receptores é referenciada através de coordenadas cartesianas. As células da matriz indicam o conjunto de pontos viáveis para posicionamento das antenas, ou seja, a região viável. A região viável constitui o espaço de busca de soluções do problema de localização de antenas. Esse espaço de busca é reduzido no instante que é definido um conjunto de pontos proibidos para posicionamento de antenas, que podem representar, por exemplo, cursos de água ou estradas. Além disto, através das variáveis de entrada do protótipo, descritas no item 3.3, o espaço de busca é delimitado pelo conjunto de pontos viáveis para localização de antenas no terreno que viabilizam o atendimento de pelo menos um ponto receptor, ou seja, locais do mapa que estiverem fora do ganho de todos receptores não farão parte do espaço de busca. Isto é possível, pois o modelo do terreno permite identificar seus pontos viáveis através de camadas (*layers*) de informação. A delimitação restringe o espaço de busca a ser utilizado pelas heurísticas de otimização, agilizando assim o processo de posicionamento das antenas.

4.2 Métodos de Otimização

Baseado no espaço de busca, delimitado pelo conjunto de pontos viáveis para posicionamento das antenas no terreno, os algoritmos de otimização realizam o processo de busca de soluções. A seguir, as estratégias adotadas na implementação dos métodos de otimização, apresentadas anteriormente no item 3.2., são descritas.

4.2.1 Solução Inicial

Ambos algoritmos de otimização, utilizados pelo protótipo, necessitam de uma solução inicial viável para iniciar o processo de busca de uma solução melhor. Uma vez definido o espaço de busca pelo conjunto dos pontos viáveis para posicionamento das antenas, admite-se que todas variáveis de entrada do sistema já foram lidas, faltando apenas estabelecer a localização inicial das antenas. Este método de geração de solução inicial é feito de duas maneiras:

- sorteio dos pontos viáveis para posicionamento das antenas;
- ou leitura de um arquivo contendo a localização inicial das antenas.

Em seguida, é feito o cálculo da função objetivo (Equação (5)) para determinar o número de receptores atendidos inicialmente.

4.2.2 Algoritmo baseado na heurística de Localização-alocação

O funcionamento deste algoritmo está baseado na criação de agrupamentos (*clusters)* de pontos viáveis para localização das antenas e posteriormente a movimentação das antenas dentro do seu agrupamento.

A partir do posicionamento inicial das antenas, definido pelo método de geração de solução inicial, são definidos *n* agrupamentos, onde *n* é o número de antenas a serem posicionadas. Os agrupamentos são formados pelos pontos viáveis para posicionamento de antenas, onde cada ponto da região viável pertence ao agrupamento da antena mais próxima. A 0 representa a formação de dois agrupamentos de pontos viáveis para o posicionamento de antenas, onde cada ponto pertence ao agrupamento da antena mais próxima.

As posições das antenas definem a solução atual do problema. O processo de movimentação das antenas consiste em trocar a posição atual de cada antena para todos pontos viáveis pertencentes ao seu agrupamento. São feitos todos movimentos de uma antena em seu agrupamento, para depois ser realizada a movimentação da próxima antena. A cada troca de posição de uma antena, o valor da função objetivo (Equação 5) é recalculado e armazenado. Ao final de todas movimentações de uma antena, o movimento que melhorou a solução atual do problema define a nova posição da antena. A nova configuração da posição de todas antenas da melhor solução encontrada passa a ser a solução atual do problema. Ao final de cada iteração, todos agrupamentos são redefinidos e caso não haja nenhuma melhora da solução atual, o algoritmo pára e a solução atual passa a ser a solução final do problema.

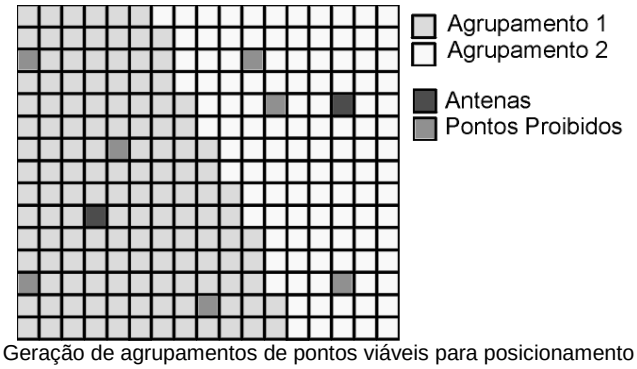


Figura 2 – Geração de agrupamentos de pontos viáveis para posicionamento de antenas

4.2.3 Algoritmo baseado em Pesquisa Tabu

Além do método de geração de solução inicial, para implementação da Pesquisa Tabu é importante definir a política de geração de vizinhanças, a Lista Tabu e o critério de aspiração.

O método de geração de vizinhanças utilizado consiste em variar as posições atuais das antenas com todas as posições do espaço de busca permitidas. Analogamente ao algoritmo de Localização-alocação, primeiro são realizadas todas trocas possíveis da posição atual de uma antena, para depois ser movimentada a próxima antena. A cada troca realizada, o valor da função objetivo (Equação 5) é recalculado e armazenado. A melhor solução viável, encontrada no conjunto de movimentos de uma antena, é conhecida como solução ótima local, e define a nova posição da antena. A nova configuração da posição de todas antenas passa a ser a solução atual do problema.

Não são admitidos movimentos que resultem numa solução do problema já visitada. Para tanto, cada solução nova encontrada é armazenada numa lista chamada de Lista Tabu. A Lista Tabu representa o conjunto de soluções que resultam a mesma configuração de receptores atendidos. Para tanto, é armazenada a lista dos receptores atendidos por cada solução ótima local encontrada. Ocorrerá um movimento proibido quando os receptores atendidos, na solução ótima local, forem os mesmos armazenados em qualquer uma das posições da Lista Tabu. Vale ressaltar que a Lista Tabu é uma lista circular, com tamanho definido pelo usuário.

Com a implementação da Lista Tabu, soluções piores do problema, em relação a solução atual, poderão ocorrer. Desta forma, a melhor solução de todas encontrada até o momento deverá ser armazenada. Caso um movimento proibido pela Lista Tabu resulte numa solução melhor que a melhor solução encontrada até o momento, o movimento é perdoado e admitido. Este mecanismo é conhecido como critério de aspiração.

Após *n* iterações sem que haja substituição da melhor solução, o algoritmo pára, retornando a melhor solução encontrada como solução do problema. O parâmetro *n*, conhecido como NBMAX, é definido pelo usuário.

4.3 Visualização de dados espaciais

A informação mais importante para o usuário consiste na localização das antenas transmissoras. Este é o resultado do processo de otimização para o qual o protótipo foi desenvolvido. Esta informação está contida nos arquivos de saída do programa, através da relação das antenas e suas respectivas localizações em coordenadas cartesianas e UTM. Todavia, para melhor visualização da disposição das antenas no terreno 3D e sua relação com os pontos receptores, foi desenvolvido um programa que utiliza OpenGL.

OpenGL é uma biblioteca gráfica para produção de aplicações que utilizam recursos gráficos em três dimensões, a partir de um conjunto de primitivas geométricas, segundo Woo *et al.* (1999). A plotagem do terreno em 3D é realizada pelo programa Gviter (programa implementado por André Detsch em disciplina de Computação Gráfica ministrado pelo professor Marcelo Walter, na Universidade do Vale do Rio dos Sinos, em 2001), no qual foram feitas modificações para suportar a visualização das antenas e dos receptores.

A Figura 3 mostra a janela de visualização de um terreno em 3D, onde os pontos representam antenas e receptores (na janela de visualização do *software* esses pontos são diferenciados por cores), e as retas, os sinais de rádio terrestres diretos entre os receptores e suas respectivas antenas servidoras. Nota-se na Figura 3-B, com cobertura, as antenas ao horizonte acima do terreno, devido sua altura adicional da torre.

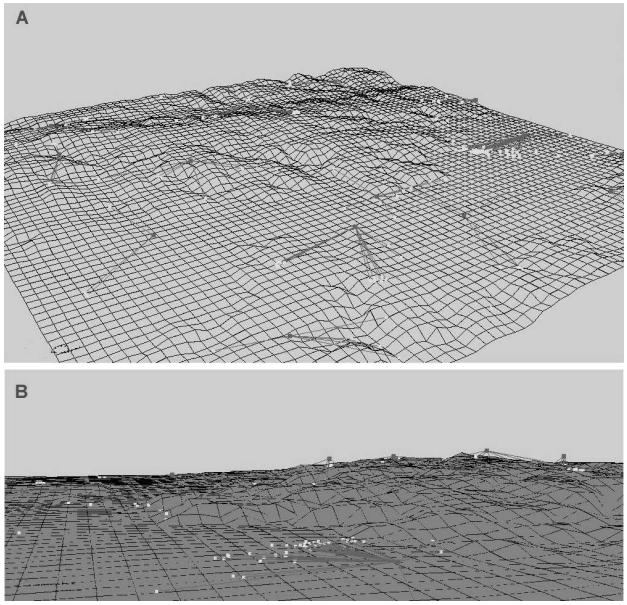


Figura 3 – Janela de visualização em 3D dos dados espaciais fornecidos pelo protótipo.

5 Experimentos

A fim de validar o protótipo desenvolvido, foram realizadas uma série de experimentos variando os dados de antenas e receptores, simulados em um equipamento Pentium III, 800 MHz, com 128 Mb de memória RAM. O protótipo foi compilado em GCC para plataforma GNU/Linux.

Nos experimentos foi utilizado um mapa da região de Tainhas- RS digitalizado de uma carta do exército brasileiro. A carta foi digitalizada com a utilização do *software* CaMap e em seguida exportados para um MDE, preservando as informações de altimetria do terreno. O mapa tem extensões de 24 por 14 quilômetros, digitalizado com 50 metros de resolução espacial, georreferenciado a um sistema de coordenadas Universal Transverso de Mercator (UTM). O posicionamento inicial das antenas, assim como a localizações dos pontos receptores foram definidas aleatoriamente, segundo uma distribuição uniforme, respeitando os limites do mapa. Nenhuma restrição quanto aos locais de posicionamento das antenas foi utilizada.

A quantidade de antenas foi variada nos seguintes números: 8 e 12. Todas com uma altura adicional em relação ao solo de 30 metros, referente a sua torre.

Da mesma forma, a quantidade de receptores variou em: 50, 100, 250 e 500. Foi considerada uma altura adicional dos receptores em relação ao solo de 2 metros.

Combinando todas variações de quantidade de antenas e receptores, totaliza-se 8 experimentos diferentes. Cada experimento foi simulado 10 vezes, com dados aleatórios.

O limite do ganho de sinal necessário para um receptor ser atendido é 0,56% do ganho de um receptor padrão comparativo, distante 1500 metros da antena, com directividade máxima. Os parâmetros do Algoritmo de Pesquisa Tabu NBMAX e tamanho da Lista Tabu foram definidos em 7 e 5 respectivamente.

6 Resultados

As Tabelas 1 e 2 sumarizam os resultados dos experimentos descritos. Cada tabela, compara os resultados dos algoritmos de otimização, variando a quantidade de receptores, sendo que nas Tabelas 1 e 2, foram utilizadas 8 e 12 antenas, respectivamente.

Ambos algoritmos de otimização chegaram a resultados com melhoras significativas a partir da solução inicial. Observa-se que, em todos experimentos, o Algoritmo de Pesquisa Tabu obteve os resultados médios superiores aos resultados do Algoritmo de Localização-alocação. Este fato deve-se a vizinhança rica gerada pelo Algoritmo de Pesquisa Tabu. Os tempos de processamento ficam significativamente longos a medida que o número de receptores e antenas aumenta, demonstrando maior robustez o Algoritmo de Pesquisa Tabu quando comparado ao Algoritmo de Localização-alocação.

Tabela 1 – Comparação de resultados de experimentos para posicionamento de 8 antenas, utilizando Pesquisa Tabu e Heurística de Localização-alocação

Qtde. Recep-tores	Solução Inicial (qtde. recep. atendidos)		Heurística	Solução Final (qtde. recep. atendidos)		Tempo processamento (segundos)	
	Média	σ		Média	σ	Média	σ
50	8,1	2,1	Tabu	34,5	2,1	75,0	4,3
			Localização	31,5	2,5	123,7	29,4
100	18,2	5,1	Tabu	58,4	2,5	188,5	10,0
			Localização	55,7	2,9	351,0	67,6
250	47,2	5,8	Tabu	119,7	4,6	624,3	157,5
			Localização	118,2	5,5	1534,9	355,3
500	94,0	11,5	Tabu	216,0	6,6	1285,4	142,4
			Localização	207,8	8,1	2585,9	524,6

σ: Desvio Padrão

Tabela 2 – Comparação de resultados de experimentos para posicionamento de 12 antenas, utilizando Pesquisa Tabu e Heurística de Localização-alocação

Qtde. Recep-tores	Solução Inicial (qtde. recep. atendidos)		Heurística	Solução Final (qtde. recep. atendidos)		Tempo processamento (segundos)	
	Média	σ		Média	σ	Média	σ
50	12,9	2,6	Tabu	42,9	1,4	96,1	5,5
			Localização	40,3	2,6	196,4	51,3
100	26,7	4,4	Tabu	75,7	2,5	257,2	23,6
			Localização	72,9	4,9	663,7	122,1
250	70,1	8,4	Tabu	162,1	4,6	956,5	295,1
			Localização	158,8	5,2	2383,9	338,8
500	138,5	11,5	Tabu	294,8	5,2	1863,75	330,4
			Localização	287,0	13,6	4659,5	968,3

σ: Desvio Padrão

7 Conclusões

Neste trabalho foram apresentados conceitos de geoprocessamento, Sistemas de Informação Geográfica e antenas, a fim de propor uma abordagem para o problema de localização de antenas. Para tanto, o problema de localização de antenas foi formulado e em seguida as heurísticas Localização-Alocação e Pesquisa Tabu foram descritas, para serem utilizadas como técnicas de otimização para do problema. Para validação da proposta, foi implementado um protótipo e realizados experimentos com dados gerados randomicamente sobre um mapa da região de Tainhas-RS. Os resultados mostraram que a arquitetura de Sistema de Informação Geográfica, utilizada para o desenvolvimento do protótipo, suportou a interpretação e manipulação de dados georreferenciados, viabilizando a análise do modelo de propagação de antenas. O SIG também facilitou a análise topológica entre as antenas transmissoras e os pontos receptores, flexibilizando o processo de geração de vizinhanças para os algoritmos de otimização de posicionamento das antenas. Observou-se que a posição inicial das antenas foi significativamente otimizada, sendo os resultados do algoritmo baseado em Pesquisa Tabu superiores quando comparados aos resultados do algoritmo baseado em Heurística de Localização-alocação. Além disto, o desempenho do algoritmo baseado em Pesquisa Tabu mostrou-se superior, resultando em tempos de processamento menores do que o algoritmo baseado na Heurística de Localização-alocação.

Os resultados mostram que o uso de SIG's é viável, para a abordagem do problema de localização de antenas. Contudo, visando à continuidade do trabalho, mais restrições podem ser consideradas no problema de modo a aproximá-lo mais à realidade. Por exemplo, outros modelos de irradiação de antenas podem ser suportados pelo protótipo, bem como a análise dos sinais de propagação terrestre por reflexão e difração.

Referências Bibliográficas

BURROUGH, A. P.; MCDONNELL, R. A. **Principles of Geographical Information Systems**. Oxford University Press, New York, 1998.

CÂMARA, G.; MONTEIRO, A.M. e CARVALHO, M.S. **Análise espacial e geoprocessamento**. In: – Câmara, G.; Monteiro, A.M.; Fuks, S.; Camargo, E.; Felgueiras,C. Análise Espacial de Dados Geográficos. São José dos Campos, INPE, 2001.

CÂMARA, G; CASANOVA, M.A.; HEMERLY, A.S.; MAGALHÃES, G.C.; MEDEIROS, C.M.B. **Anatomia de Sistemas de Informações Geográficas**. 10º Escola de Computação, Instituto de Computação, UNICAMP, Campinas, SP, 1996.

CÂMARA, G. **Modelos, Linguagens e Arquiteturas para Bancos de Dados Geográficos**. Tese de Doutorado, INPE, São José dos Campos, SP, 1995.

CARVALHO, M.S.; PINA, M.F.; SANTOS, S.M. **Conceitos Básicos de Sistema de Informações Geográficas e Cartografia Aplicados à Saúde**. Organização Panamericana da Saúde / Ministério da Saúde, Brasília, 2000.

COOPER, L. **Location-allocation problems**. Operation Research, 11:331-343, 1963.

DAVIS, C. ; CÂMARA, G. **Arquitetura de sistemas de informação geográfica.** In: – Câmara, G.; Monteiro, A.M.; Fuks, S.; Camargo, E.; Felgueiras,C. Análise Espacial de Dados Geográficos. São José dos Campos, INPE, 2001.

EASTMAN, J.R. **Idrisi for windows versão 2 - Manual do usuário**. Clark Labs for Cartographic. Edição em português por Hasenack, H. e Weber, E., Centro de Recursos Idrisi no Brasil, Porto Alegre, 1998. Disponível em http://www.ecologia.ufrgs.br/idrisi, acessado em 15-set-2002.

GALVÃO, R.D.; REVELLE, C. **A Lagrangean heuristic for the maximal covering location problem**. European Journal of Operational Research, 88:114-123, 1996.

GALVÃO, R.D.; ESPEJO, L.G.A.; BOFFEY, B. **A comparison of lagrangean and surrogate relaxations for the maximal covering location problem**. European Journal of Operational Research, 124:337-389, 2000.

GLOVER, Fred. **Tabu Search I**. Colorado - University of Colorado, 1989.

GLOVER, Fred; LAGUNA, Manuel. **Tabu Search**. Kluwer Academic Publishers, 2001.

GOLDBARG, M.C.; LUNA, H.P.L. **Otimização Combinatória e Programação Linear: modelos e algoritmos**. Campus, Rio de Janeiro, 2000.

HERTZ, A. **Tabu Search for Large Scale Timetabling Problems.** Lausanne - Ecole Polytechnique Fédérale de Lausanne, 1991.

INPE, **''Tutorial 10 aulas - Spring 3.5.1''**. INPE - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, 2001.

LEE, R.C.; TEPFENHART, W.M. **UML e C++ - Guia Prático de Desenvolvimento Orientado a Objeto**. Mkron Books, São Paulo, 2001.

LORENA, L.A.N. **Análise de redes**. In: Câmara, G.; Monteiro, A.M.; Fuks, S.; Camargo, E.; Felgueiras,C. Análise Espacial de Dados Geográficos. São José dos Campos, INPE, 2001.

LORENA, L.A.N. ; SENNE, E. L. F. ; PAIVA, J. A. C. e PEREIRA M. A.: **Integração de modelos de localização a sistemas de informações geográficas**. Gestao e Produção 8(2):180-195, 2001.

LYTEL, A.: **ABC das Antenas**. Antenna Empresa Jornalística S.A., Rio de Janeiro, 1973.

MAGUIRE, D.J.; GOODCHILD, M.F.; RHIND, D.W. **Geographical Information Systems – volume II**. John Wiley & Sons, 1991.

MARTINHA, A.V. **Antenas**. Editorial Presença, Lisboa, 1987.

MURAI, S. **SIG Manual Básico: Volume 1: Conceptos Fundamentales**. Revista Selper, Vol. 15, nº 1, Universidade de Tokio, Japão, 1999.

PAPADIMITRIOU, C.H.; STEIGLITZ, K. **Combinatorial Optimization – Algorithms and Complexity**. Dover, Mineola, 1998.

PIDD, M. **Modelagem Empresarial – Ferramentas para tomada de decisão**. Bookman, Porto Alegre, 1998.

RARDIN, R.L. **Optimization in Operations Research**. Prentice Hall, New Jersey, 1998.

ROSSITER, D.G. **Land Evaluation - Parte 2: Geographical Information Systems.** Cornell University - College of Agriculture & Life Sciences, Lecture Notes, 1 – 31, 1994.

SCAPARRA, M.P.; SCUTELLÀ, M.G. **Facilities, locations, customers: building blocks of location models. A survey**.. Technical Report TR-01-18, Università de Pisa, Pisa, 2001.

SMIT, J. **Rádio Propagação**. Érica, São Paulo, 1986.

TAILLARD, E.D. **Heuristic methods for large centroid clustering problems**. Technical Report IDSIA96-96, IDSIA, 1996.

VASSALO, F.R. **Manual de Antenas Receptoras para TV e FM**. Plátano Editora, Lisboa, 1979.

WOO, M.; NEIDER, J.; DAVIS, T.; SHREINER, D. **OpenGL – Programming Guide**. Addison-Wesley, Massachusetts, 1999.

Aplicação que usa Protocolo de Perfil Leve para Transferência de Arquivos

Cleber Machado Ortiz (UFRGS)
ortiz@inf.ufrgs.br

Lisandro Zambenedetti Granville (UFRGS)
granville@inf.ufrgs.br

Liane Margarida Rockenbach Tarouco (UFRGS)
liane@inf.ufrgs.br

Resumo. Videoconferência contempla, além do intercâmbio de áudio e vídeo entre duas pessoas ou grupos, também o compartilhamento de dados. Sistemas que permitem a colaboração de dados são usualmente integrados aos sistemas de videoconferência e no cenário atual a padronização está sendo feita a partir de recomendações definidas pelo ITU nas séries H.323 e T.120. Os serviços e protocolos definidos nas recomendações H.323 (para videoconferência) e T.120 (para a colaboração de dados) do ITU são bastante complexos e ocorrem muitos problemas na sua utilização em redes de pacotes funcionando segundo o princípio de best effort da Internet. Com vistas a superar os problemas usualmente encontrados, foi feita uma análise desta padronização e de implementações típicas com vistas a propor uma simplificação capaz de assegurar as funcionalidades previstas na T.120 mas que tenha menor complexidade.

Palavras-chave: Videoconferência, Colaboração Visual de Dados, Transferência de Arquivos, T.120, T.127, Perfil Leve, H.323

1 Introdução

Diversos avanços das tecnologias que surgiram no final do século passado foram essenciais para a obtenção, compartilhamento e disseminação da informação. A respeito dessa informação, hoje em dia é possível agregar alguns fatores importantes para a comunicação através de redes de computadores, tais como áudio, vídeo e dados, sendo que a união desses termos ou elementos pode ser considerada como multimídia.

Uma grande vantagem das soluções e recursos da videoconferência é o fato de pessoas geograficamente distantes poderem interagir através de um computador pessoal, eliminando com isso despesas e a perda de tempo inerente aos deslocamentos “tradicionais”. Buscando tais vantagens, muitas empresas começam a buscar soluções passíveis de utilização num contexto mais abrangente, com um número maior de produtos e infraestruturas, investindo mais em soluções padronizadas em lugar de soluções proprietárias. Atualmente uma grande variedade de soluções para sistemas de videoconferência está disponível e cada aplicação, de acordo com o seu fornecedor, pode ter necessidades diferentes com relação a equipamentos, à infra-estrutura de comunicação e à qualidade de serviço (Videoconferencing 2002). Considerando estes fatores, um sistema de videoconferência deve se adequar da melhor forma possível aos recursos que a infra-estrutura de rede oferece (Leopoldino e Moreira 2002).

O padrão H.323 do ITU (ITU-A 1998) estabelece um conjunto de serviços e protocolos para assegurar a interoperabilidade entre clientes e servidores de videoconferência nas trocas de áudio, vídeo e controle da videoconferência. Mas durante uma videoconferência é natural a necessidade de compartilhamento de dados na forma de mensagens (*chat*), arquivos, estruturas compartilhadas tais como quadro-branco e também permitir que uma aplicação, sendo executada na estação de trabalho de um dos participantes, seja acompanhada ou mesmo controlada por um outro participante (compartilhamento de aplicações). Esta classe de serviços foi objeto de padronização

pelo ITU e sua especificação publicada na série de recomendações conhecida como T.120 (ITU-B 1996).

Estas aplicações usam protocolos bastante complexos que contemplam o objetivo de permitir que ocorra um ajuste nos serviços oferecidos tendo em vista as condições de transmissão, derivadas de condições tanto do enlace como da própria estação cliente do serviço. Visando investigar uma possível solução que preserve as funcionalidades essenciais mas que permita reduzir a sobrecarga (tráfego de mensagens de controle e processamento) inerente ao serviço T.120, foi realizada uma análise teórica (a partir das normas ITU-T) e experimental (monitoração de tráfego e desenvolvimento de protótipo) visando delinear uma proposta de serviço equivalente mas com características de menor complexidade .

O objetivo deste artigo é apresentar alguns resultados do estudo realizado sobre o contexto de videoconferência, suas soluções e protocolos com ênfase nos protocolos padronizados para colaboração de dados dentro de uma videoconferência (mais conhecida como padronização T.120) e a sua estrutura. Baseado nestes estudos, o trabalho apresenta uma proposta de solução para uma aplicação que utilize transferência de arquivos nos moldes do padrão ITU T.127 (ITU-C 1995), mas que atenda aos requisitos de menor complexidade (tráfego e processamento) . Esta proposta utiliza as estratégias de simplificação dos protocolos usados no contexto de colaboração de dados em ambiente que ainda mantenha compatibilidade com o ambiente T.120 (ITU-D 1995).

O restante do artigo está assim dividido: na seção 2, é apresentada um estudo sobre os sistemas de videoconferência, o tópico 3, dá uma idéia sobre as padronizações encontradas na videoconferência bem como sua arquitetura. Na seção 4, são apresentados os protocolos auxiliares para colaboração de dados e sua arquitetura conforme o ITU. No tópico 5, é apresentada a arquitetura para transferência de arquivos. A seção 6 apresenta a proposta minimalista para o protocolo de transferência de arquivos T.127. Por fim, no seção 7 são feitas as considerações e trabalhos futuros.

2 Sistemas para videoconferência

Os sistemas de videoconferência permitem que se trabalhe de forma cooperativa e se compartilhe informações e materiais de trabalho sem a necessidade de locomoção geográfica. Através da videoconferência, são criadas novas possibilidades para reuniões de pessoas distantes umas das outras, provendo diversos meios bem mais interessantes pela comunicação visual e a interação entre os demais participantes, superando desta forma o conhecido e-mail, os diversos canais de bate-papo e até mesmo o telefone.

A proposta contida no padrão ITU H.323 (ITU-A 1998) especifica componentes básicos dos serviços de videoconferência tais como MCUs (*Multipoint Control Unit*), terminais, *gatekeepers* e *gateways*. A seção 3.1 contém um maior detalhamento em relação a estes componentes. Sobre os tipos de interação em videoconferência, existem diversos sistemas disponíveis (Leopoldino 2002), classificados segundo a forma de comunicação que utilizam:

- Ponto-a-ponto: quando existe a comunicação entre duas pessoas ou mais.
- Modelo centralizado: Este modelo é baseado no modo de comunicação ponto-a-ponto ou *unicast.* No caso de existirem três ou mais pontos para se conectarem entre si, a comunicação é possível desde que haja um MCU, ou Unidade de Controle Multiponto.
- Modelo descentralizado: O modelo descentralizado compartilha características de controle comum com o modelo centralizado, mas o fluxo de mídia é manuseado diferentemente. Uma das entidades participantes deve ser um MC (Controlador Multiponto) que, independente do modelo de comunicação, provê o controle de três ou mais participantes durante uma sessão multiponto. Não existe MCU para processar os

múltiplos fluxos; cada participante é responsável por sua própria mesclagem de áudio e seleção de vídeo.

- Modelo híbrido: O modelo híbrido tenta mesclar o melhor dos dois modelos anteriores, mantendo a consistência dos dados através de um armazenamento centralizado e suportando visões individualizadas através do uso de *front ends.*

3 Padronização do serviço de videoconferência

Em uma fase inicial, coexistiram soluções usando protocolos proprietários. A solução preponderante no cenário atual é a da recomendação H.323 do ITU, que descreve uma arquitetura contendo terminais, equipamentos e serviços para comunicação multimídia em uma ou mais redes locais, LAN's, MAN's e WAN's incluindo a Internet (sem garantia da qualidade no serviço). Na verdade, a H.323 é uma recomendação "guarda-chuva", referenciando outras recomendações ITU-T e até o protocolo RTP *(Real-Time Transport Protoco*l) do IETF. A primeira versão desta recomendação foi publicada em 1996 e a segunda em 1998 (ITU-A 1998). Esta última contém melhorias no tocante à segurança, diminuição do tempo necessário para liberação do canal após a chamada ter sido atendida pelo destino, incorporação de serviços adicionais como transferência e redirecionamento, e maior integração com T.120 (protocolo de transmissão de dados multimídia).

Componentes da arquitetura H.323

O ITU-T define na recomendação H.323 a descrição dos seus componentes, visando prover serviços de comunicação multimídia sobre redes de pacotes. Os componentes descritos compreendem terminais, *gateways*, *gatekeepers* e MCUs (Figura 1).

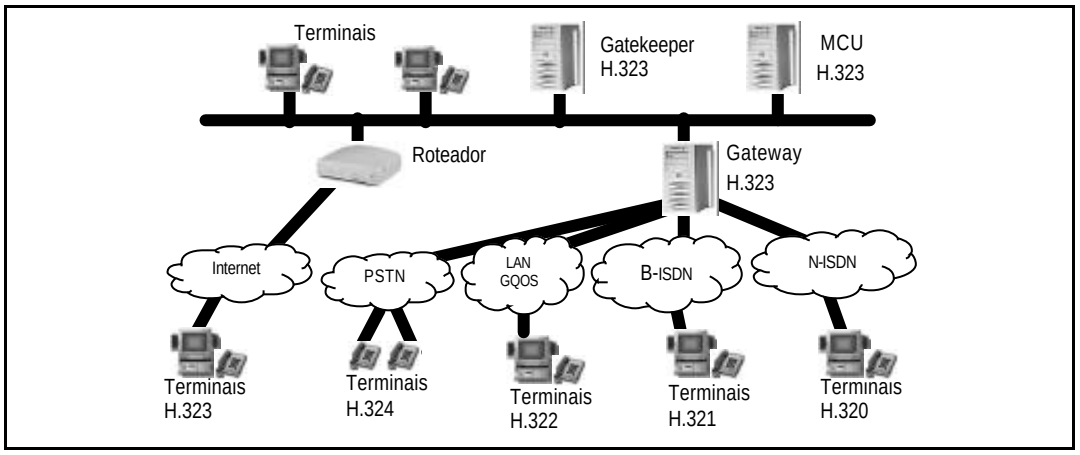


Figura 1: Componentes da arquitetura H.323

- Terminais: Por definição é um ponto de rede que provê comunicação bidirecional, em tempo real, com outro equipamento terminal, *gateway* ou MCU.
- Gateway: Converte, apropriadamente, diferentes formatos de mensagens e procedimentos de comunicação. A conversão entre diferentes formatos de áudio, vídeo ou dados também pode ser feita pelo *gateway*.
- Gatekeeper: É um elemento opcional no sistema H.323, provendo serviços de controle de chamadas para os terminais.

- MCU: O MCU é um componente do H.323 que provê a capacidade para suportar conferências multiponto. Pode estar dividido em um MC (*Multipoint Controller* - obrigatório) e MPs (*Multipoint Processor* - não obrigatório) [7].

Sinalização no H.323

A sinalização H.323 é complexa (ITU-A 1998), devido principalmente à sua extensa pilha de protocolos, e a necessidade de conformidade com padrões antigos da ITU-T. Como exemplos de mensagens intercambiadas em uma videoconferência, ilustradas na figura 2, pode-se referir:

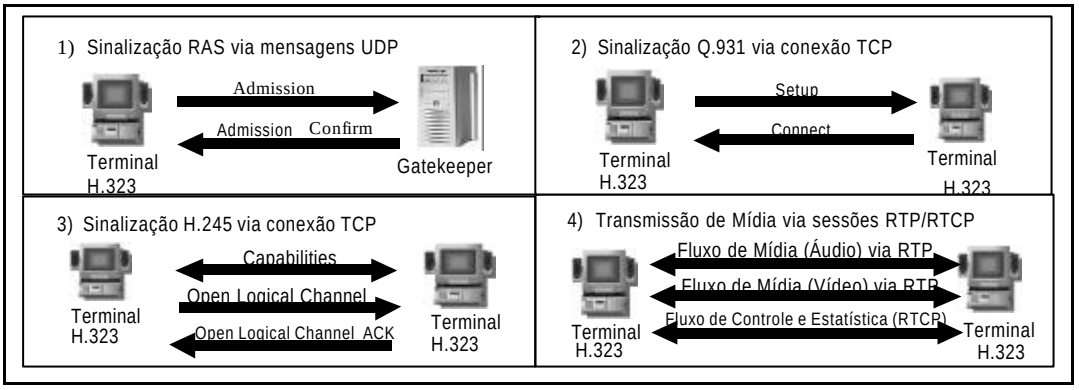


Figura 2: Sinalização no H.323

- Mensagens ARQ (*Admission Request*, ou pedido de abertura de sessão), ARJ (*Admission Reject*) e ACF (*Admission Confirm*) que são exclusivas dos terminais H.323. Um terminal registrado em um *gatekeeper* pede autorização ao gatekeeper para iniciar e/ou aceitar chamadas.
- Mensagens Q.931 são SETUP (estabelecimento de chamada), *Call Proceeding* (processamento da chamada ou solicitação) e CONNECT (confirmação do estabelecimento de chamada).
- Na fase de estabelecimento da comunicação entre terminais ou entre terminais e MCU existe um procedimento de inicialização de mídia definido pela recomendação H.245. Nesta etapa, uma porta TCP é aberta para negociação dos subconjuntos de mídias suportados e a ordem de preferência das mídias. O canal H.245 é mantido aberto caso alguém abra uma nova sessão de mídia, ou modifique uma existente. As mensagens mais básicas do H.245 são: *Capability Exchange* (troca de conjuntos de capacidades de mídia entre os terminais), *Open Logical Channel* (abertura de canal de controle do fluxo de mídia) e *Open Logical Channel Acknowledge* (confirmação do mesmo).
- O transporte do fluxo de mídia, após a fase de negociação, acontece no nível de rede pelo uso do protocolo de transporte RTP (*Real time Transport Protocol*) .

4 Protocolos para a colaboração de dados

A recomendação T.120 contém uma série de recomendações que oferecem suporte para o estabelecimento de conferências multimídia de dados em uma arquitetura multiponto e em tempo real. Esta recomendação, apresenta um conjunto de protocolos de comunicação e aplicação, que são coletivamente referenciados como a Série T.120.

A padronização T.120 descreve as recomendações que compõe a serie. Relata como as recomendações se relacionam entre si para o estabelecimento de conferências e ainda, a arquitetura

proposta para a troca de dados em um ambiente de conferência multimídia. Dentre as recomendações da Serie T.120, pode-se destacar (ITU-B 1996):

- Recomendação ITU-T T.121 – (1996) – Modelo genérico de aplicação *(Generic application template)*
- Recomendação ITU-T T.122 – (1996) – *Serviço de comunicação multiponto para a definição de serviços de conferência audiovisual e audiográficas (Multipoint communication service for audiographics and audiovisual conferencing service definition)*
- Recomendação ITU-T T.123 – (1996) – *Pilha de protocolos para aplicações de teleconferência audiovisual e audiográficas (Protocol stacks for audiographic and audiovisual teleconference application)*
- Recomendação ITU-T T.124 – (1996) – *Controle de conferência genérico (Generic conference control)*
- Recomendação ITU-T T.125 – (1996) – *Especificação do protocolo de serviço de comunicação multiponto (Multipoint communication service protocol specification)*
- Recomendação ITU-T T.126 – (1996) – *Protocolo de anotação e de imagens estáticas Multiponto (Multipoint still image and annotation protocol)*
- Recomendação ITU-T T.127 – (1996) – *Protocolo para transferência multiponto de arquivo binário (Multiponit binary file trasfer protocol)*
- Recomendação ITU-T T.128 – (1996) – *Compartilhamento de aplicações multiponto (Multiponit application sharing)*

Os protocolos que compõem a Serie T.120, possuem a capacidade de manipular mais de uma conferência ao mesmo tempo, assim como, um determinado terminal pode participar em mais de uma conferência.

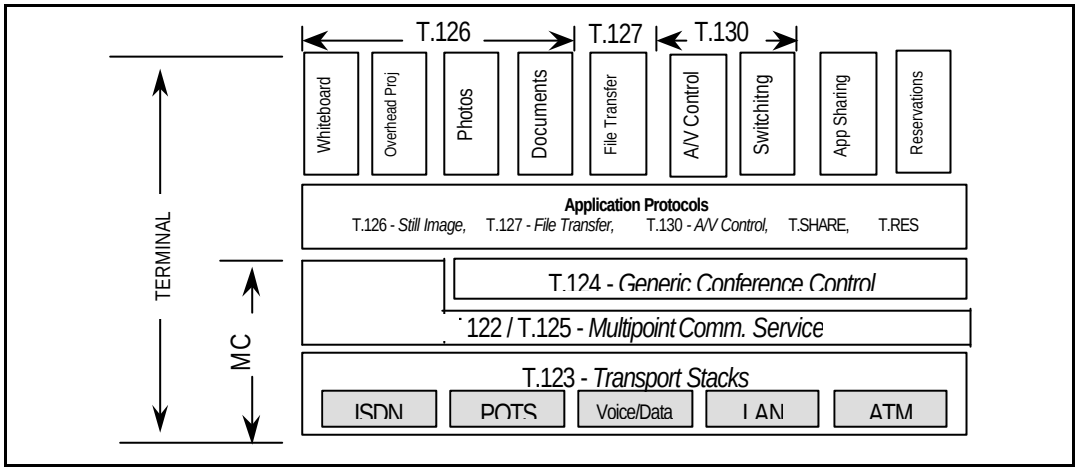


Figura 3: A arquitetura da recomendação T.120

A recomendação T.120 contempla a capacidade de serem manipuladas diferentes taxas de fluxo de informação numa mesma conferência, dependendo dos limites impostos por cada uma das tecnologias de rede usadas pelos que participam da conferência. Em se tratando de troca de informações multiponto através de uma conferência T.120, apenas as recomendações T.122, T.123, T.124, T.125 são obrigatórias e devem ser implementadas nos softwares ou dispositivos de

conferência T.120. A figura 6 apresenta um diagrama dos módulos integrantes da arquitetura T.120.

Os nós participantes de uma conferência T.120 devem estar organizados hierarquicamente e deve existir somente um nó no topo da árvore. A topologia na qual a conferência pode estar organizada pode variar entre, estrela ou cadeia de nós ou uma combinação destas duas, entretanto a comunicação entre os terminais não pode apresentar redundâncias (*loops*) e o nó que está no topo da conferência deve estar presente desde o início da conferência até o seu final.

A padronização trata a conexão ponto-a-ponto como a forma mais simples de conexão multiponto e define algumas possibilidades para o estabelecimento de conexões multiponto. Por exemplo, terminais com múltiplas portas, onde cada porta deve implementar ao menos os protocolos especificados nas recomendações T.122, T.123, T.124 , T.125, que são obrigatórias. Neste caso, este terminal pode atuar como uma ponte entre três ou mais nós permitindo que uma comunicação multiponto possa ser estabelecida entre estes nós.

MCS

O serviço de comunicação multiponto (MCS) está definido nas recomendações T.122 e T.125 da série T.120. O MCS trata-se de um serviço genérico, projetado para suportar a comunicação multiponto *full-duplex* entre um número arbitrário de entidades de aplicação, sobre uma variedade de tecnologias de rede, conforme especificado na recomendação T.123. A recomendação T.122 define os serviços multiponto disponíveis aos desenvolvedores, a recomendação T.125, o protocolo para a transmissão dos dados.

O MCS executa várias funções chaves para que a troca de dados possa ocorrer em uma conferência, como por exemplo: gerenciamento de domínios, gerenciamento dos canais, transferência de dados, gerenciamento de “tokens” e notificação das capacidades. A recomendação T.122 define todos os serviços que deverão ser oferecidos pelo MCS, às aplicações (usuários de MCS). A recomendação define primitivas de serviço (variações, tipos e parâmetros), além da seqüência em que deve ser as mensagens do protocolo devem ocorrer entre os provedores de MCS. Também pode definir os valores mínimo e máximo destes parâmetros.

Para controlar os múltiplos domínios que podem coexistir em um MCS, ele mantém uma base de informações para o gerenciamento dos recursos de MCS que estão sendo utilizados pelos domínios, em um dado instante. O estado dos canais e dos tokens em uso naquele domínio em um determinado momento e as informações registradas nas bases de informações são recursos a serem gerenciados.

O padrão T.125 especifica o formato das mensagens do protocolo e os procedimentos necessários para a troca de mensagens sobre um conjunto de conexões de transporte. A finalidade do protocolo é implementar os serviços de comunicação multiponto definidos pela recomendação T.122. O conjunto dessas duas recomendações define o MCS. A especificação T.125 descreve os procedimentos de um protocolo responsável por realizar a transferência de dados e informações de controle entre as entidades que provêm o MCS e que fazem parte de um domínio. O T.125 também é responsável por definir uma estrutura e codificação dos pacotes (PDU - unidades de dados de protocolo) usados na comunicação.

O GCC

A padronização T.124 é responsável por definir o Controle Genérico de Conferência (GCC). Esta estrutura é voltada para o gerenciamento e controle dos terminais audiográficos ou audiovisuais e de MCUs. O T.124 juntamente com as recomendações T.122, T.123 e T.125, constituem um conjunto mínimo de recomendações da série T.120 para o desenvolvimento de terminais ou MCUs. A recomendação T.124 define os seguintes componentes funcionais que

formam o GCC: estabelecimento e encerramento de conferências; gerenciamento da lista da conferência; gerenciamento da lista de aplicações e serviços de registro de aplicações e condução da conferência.

A recomendação define os serviços relacionados com as primitivas dos componentes citados anteriormente. Este padrão também mostra o protocolo e os PDUs associados a estes serviços do GCC. Este protocolo e os PDUs são essenciais na comunicação entre o GCC e o MCS. Numa conferência deve necessariamente existir a figura do Provedor principal de GCC, que pode ser o próprio MCU da conferência e que irá centralizar as informações necessárias para os GCCs presentes na conferência. A figura 4 ilustra este cenário.

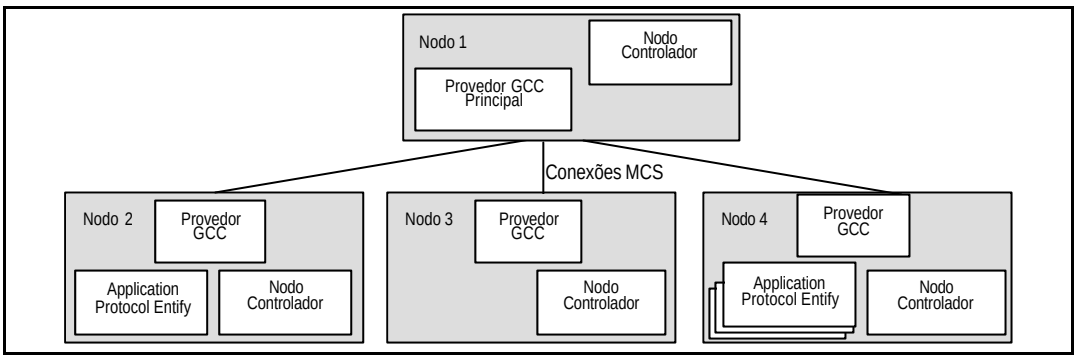


Figura 4: Componentes do GCC em um domínio MCS

O padrão GCC oferece alguns serviços para estabelecer e encerrar conferências. Entre esses serviços: serviços para encontrar quais conferências estão acontecendo, entrar em uma conferência, deixar uma conferência, restringir o acesso a uma conferência, entre outros são oferecidos para as aplicações.

Os participantes devem saber quais as informações são necessárias para que ele possa ingressar na conferência. O conjunto de serviços para o estabelecimento e encerramento de conferências oferece um mecanismo para que os participantes possam ver o nome das conferências em andamento e selecionar a conferência que ele deseja se juntar. Na criação de uma conferência, as características da conferência, conhecidas como perfis da conferência são especificadas pelo seu criador. Um método para a criação de conferências também é oferecido pelo GCC.

5 Arquitetura para transferência de arquivos T.127

A Recomendação T-127, proposta pela ITU-T, define um protocolo para suporte a troca de arquivos binários dentro de uma conferência interativa de grupo, trabalhando em um ambiente onde o T.120, é usado (ITU-C 1995). Provê mecanismos para suporte à distribuição simultânea de múltiplos arquivos, distribuição seletiva de arquivos para um subconjunto de participantes, recuperação de arquivos de locais remotos, permitindo, também, acesso a diretório remoto, usam primitivas previstas pela Recomendação T.122 (*Multipoint Communications Service*). T.127 foi projetado para oferecer um protocolo que provê um conjunto funcional para permitir interconexão entre aplicações que requerem uma capacidade de transferência de arquivo básica e também tem a flexibilidade para conhecer as demandas de aplicações mais sofisticadas.

T.127 dá suporte a dois tipos de canal de dados: *broadcast* e *acknowledged*. Se um transmissor deseja enviar a todos os nodos um arquivo que está oferecendo, então deverá usar o canal de dados de *broadcast*. Todos os nodos têm que ficar ligados ao *canal broadcast de dados* durante a sessão de transferência do arquivo e é obrigatório receber todos os arquivos distribuídos neste canal; se um arquivo não é requerido, os receptores deverão descartá-lo (Leopoldino 2002).

No caso de um transmissor desejar dar para outros nodos a opção de rejeitar um arquivo, deverá oferecer o arquivo em um canal de dados *acknowledged*. Neste caso, cada nodo tem que informar ao transmissor se deseja ou não os dados, e só esses que desejam o arquivo é que se ligam ao canal de dados. Transferências de múltiplos arquivos simultaneamente são suportados pelos canais de dados *acknowledged*s.

Deverão ser usados canais de dados *acknowledged*s se um transmissor considera os parâmetros do header de arquivo essenciais à operação da aplicação. Por exemplo, uma aplicação pode exigir preservar um *pathname* por receptores para referência futura. São identificados parâmetros chaves quando são oferecidos arquivos para distribuição; nodos que estão impossibilitados de suportar todos os parâmetros têm que rejeitar o arquivo. O criador de um canal de dados *acknowledged* pode designá-lo para uso exclusivo (só o criador pode enviar arquivos no canal) ou compartilhado (i.e. qualquer participante pode enviar arquivos nele).

Transações com arquivos no canal de *broadcast* não requerem nenhum *handshaking* (estabelecimento de acordo) entre transmissor e receptores como nodos obrigados a receberem todos os arquivos distribuídos neste canal. Isto minimiza a latência no começo de transferências de arquivo para transações no canal de dados de *broadcast*. Transações no canal de dados *acknowledged* provoca alguma latência no começo de uma transferência de arquivo, mas pode ter um desempenho global melhor evitando distribuição desnecessária de dados para locais que não estão desejando recebe-lo, particularmente em locais com largura de banda baixa. A escolha de canal está na discrição do transmissor e pode depender da aplicação, tamanho do arquivo, configuração da rede e número de participantes da conferência. Aplicações que desejam dar suporte ao protocolo de transferência de arquivo devem estar aptas a se unir ao canal de controle e enviar ou receber no canal de dados de *broadcast*.

Aplicação no T.127

Uma transferência de arquivo pela aplicação do usuário confia nos serviços de uma Entidade de Protocolo de Aplicação para Transferência de Arquivo (File Transfer Application Protocol Entity -*File APE*) que se comunica com aplicações de outros nodos. A *File APE* tem dois componentes: *File Transfer Application Resource Manager* (File ARM) e um *File Transfer Application Service Element* (File ASE) (figura 5).

O ARM provê funcionalidade genérica, comum a todos os protocolos de aplicação, além disso, o ASE provê funcionalidade específica para este protocolo de aplicação habilitar interconexão de aplicações para transferência de arquivo.

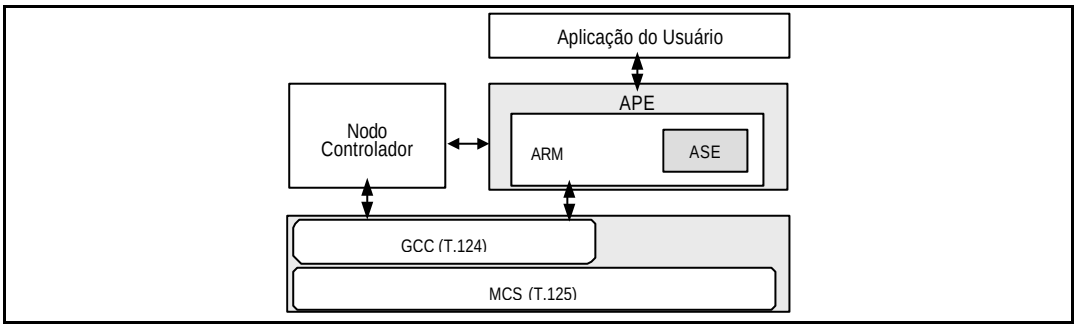


Figura 5: Modelo de aplicação T.127

6 Proposta Minimalista para o Protocolo de Transferência de Arquivos T.127

O principal motivo de criar um modelo leve (*lightweght* T.120) para o T.127, é o fato de que implementações que trabalham em cima da pilha T.120 apresentam uma grande complexibilidade, seja esta na comunicação entre camadas como GCC e MCS, seja na própria comunicação com o T.127 (ITU-D 1995). Outro fator importante e muito relevante é relativo à capacidade de poder controlar ou limitar a quantidade de largura de banda que é alocada para uma conferência.

Na figura 6, é ilustrada a sinalização derivada de problemas que ocorrem com freqüência maior do que o aceitável em conferência com colaboração de dados. Ali é indicado o encerramento ocasionado em uma videoconferência por banda insuficiente. Na sessão ilustrada na figura 6, foram solicitados serviços auxiliares para colaboração de dados (como a transferência de arquivos). O T.120 trabalha utilizando toda ou boa parte dos recursos da rede destinados a uma conferência, quando um recurso solicita uma capacidade além do que está sendo utilizado na conferência, esta deve ser finalizada com a notificação de banda insuficiente.

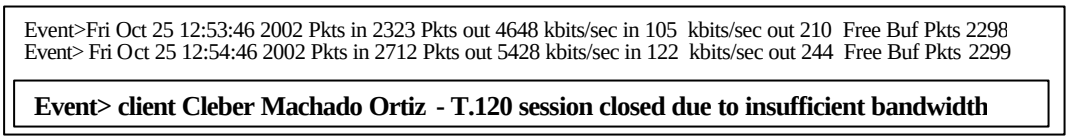

Event>Fri Oct 25 12:53:46 2002 Pkts in 2323 Pkts out 4648 kbits/sec in 105 kbits/sec out 210 Free Buf Pkts 2298
Event> Fri Oct 25 12:54:46 2002 Pkts in 2712 Pkts out 5428 kbits/sec in 122 kbits/sec out 244 Free Buf Pkts 2299
Event> client Cleber Machado Ortiz - T.120 session closed due to insufficient bandwidth

Figura 6: Conferência encerrada por banda insuficiente

A partir da análise destas e de outras trocas, foi formulada uma primeira hipótese de simplificação para o serviço que envolve limitar o número e tipo (em função da demanda de capacidade que podem requerer) de chamadas entre as camadas de uma arquitetura e conseqüentemente simplificar a implementação. Pode-se ter um protocolo mais "leve" para trabalhar sobre um ambiente de colaboração visual de dados, e que não utilize toda ou boa parte capacidade disponível da rede, mas que sempre use um valor fixo.

Um conjunto de estratégias para um perfil T.120 minimalista, simplificando um nodo T.120 tradicional pode ainda envolver restrição nas operações seguintes (ITU-D 1995):

a) Suporte a somente uma única conexão por nodo;

b) Participação em uma única conferência de cada vez (exemplo, uma conexão de MCS, um domínio de MCS, uma única hierarquia de protocolo de transporte, uma conferência de GCC entre outros);

c) Uso de somente uma única entidade de protocolo de aplicação (APE);

d) Restringir o provedor principal para conferências ponto-a-ponto entre dois nodos que requerem funcionalidades de um provedor.

Um Protocolo Leve de Transporte da Rede (T.123)

No que diz respeito aos parâmetros do protocolo Q.922, o desenvolvedor é livre para adaptar os parâmetros sugeridos pela recomendação T.123. Por exemplo, quando o terminal sabe qual a priori qual a aplicação que usa algum tipo de controle, como por exemplo, compartilhamento de aplicações (T.128), os parâmetros do Q.922 podem ser simplificados.

Em particular, terminais de áudio/vídeo com pouca largura de banda que só implementam serviços T.120 para estações T.132, podem tirar proveito desta possibilidade e reduzir os parâmetros do protocolo Q.922.

A Recomendação T.122/T125 com Características Minimalistas

No que diz respeito aos parâmetros do protocolo Q.922, o desenvolvedor é livre para adaptar os parâmetros sugeridos pela recomendação T.123. Por exemplo, quando o terminal sabe qual a priori qual a aplicação que usa algum tipo de controle, como por exemplo, compartilhamento de aplicações (T.128), os parâmetros do Q.922 podem ser simplificados.

Em particular, terminais de áudio/vídeo com pouca largura de banda que só implementam serviços T.120 para estações T.132, podem tirar proveito desta possibilidade e reduzir os parâmetros do protocolo Q.922.

Tabela 1: Primitivas do MCS para uma solução minimalista

Unidade Funcional	Primitivas	MCSPDUs associados
Domain Management	MCS-CONNECT-PROVIDER request	ConnectInitial
	MCS-CONNECT-PROVIDER indication	ConnectInitial
	MCS-CONNECT-PROVIDER response	ConnectResponse
	MCS-CONNECT-PROVIDER confirm	ConnectResponse
	MCS-DISCONNECT-PROVIDER request	DisconnectProviderUltimatum
	MCS-DISCONNECT-PROVIDER indication	DisconnectProviderUltimatum
	MCS-ATTACH-USER request	AttachUserRequest
	MCS-ATTACH-USER confirm	AttachUserConfirm
	MCS-DETACH-USER request	DetachUserRequest
	MCS-DETACH-USER indication	DetachUserIndication
Channel Management	MCS-CHANNEL-JOIN request MCS-CHANNEL-JOIN confirm	ChannelJoinRequest ChannelJoinConfirm
	MCS-CHANNEL-LEAVE request	ChannelLeaveRequest
Data Transfer	MCS-SEND-DATA request	SendDataRequest
	MCS-SEND-DATA indication	SendDataIndication
	MCS-UNIFORM-SEND-DATA request	UniformSendDataRequest
	MCS-UNIFORM-SEND-DATA indication	UniformSendDataIndication

O Perfil T.124

T.124 descreve as primitivas de serviços solicitados de um provedor T.124 para um nodo T.120 minimalista. Para esta parte do perfil, é solicitado somente um subconjunto dos serviços do GCC. Estes serviços incluem, estabelecimento de conferência e término, suporte para a lista de aplicações na conferência (*application roster*), suporte para uma única aplicação na lista.

Em caso de chamadas a PDUs, estes podem ser ignorados ou não. No caso de serem ignorados, elas não requerem nenhum comportamento default quando recebidos. Um PDU deste tipo pode ser recebido a qualquer momento e deve ser decodificado pelo menos no nodo receptor antes de ser descartado.

O Perfil T.121

O T.121 descreve os serviços requeridos por um provedor T.120 que implementa o Modelo de Aplicação Genérico (T.GAT ou T.121) para um nodo T.120 minimalista. Para este perfil, é necessário somente um subconjunto dos serviços do GAT. Estes perfis só incluem serviços associados como criação de uma sessão.

Com isso, são suportados somente canais estáticos e tokens dentro do perfil T.120 do MCS e nenhum GCC suporta funções relacionadas com o registro.

Chamadas a partir do T.127 que podem ser minimalizadas

Com o objetivo de implantar um serviço auxiliar de colaboração visual de dados que contenha um perfil minimalista, são sugeridas algumas abstrações ou "podas" dentro da padronização. Essas "podas" não podem impedir o funcionamento de uma estrutura já definida, mas diminuir a complexibilidade. Tendo então uma aplicação que trate da transferência de arquivos sobre T.120, com características minimalistas (Tabela2).

Tabela 2: Chamadas que podem ser minimalizadas na padronização T.120

Chamadas	Considerações
Private-Channel-Join-InvitePDU (Private-Channel-Join-InvitePDU)	**Recepção de arquivos apenas para APE:** - Chamada deste tipo, pode ser opcional no envio de PDU; - No caso do recebimento do PDU, torna-se uma chamada obrigatória; **Transmissão de arquivos apenas para APE:** - Esta chamada pode ser opcional no envio de PDU; - Para o recebimento do PDU, torna-se uma chamada obrigatória; **Transmissão e recepção de arquivos para APE:** - Esta chamada pode ser opcional no envio de PDU; - No caso do recebimento do PDU, torna-se uma chamada obrigatória;
private-Channel-Join-ResponsePDU (Private-Channel-Join-ResponsePDU)	**Recepção de arquivos apenas para APE:** - Esta chamada pode ser obrigatória no envio de PDU; - Para o recebimento do PDU, torna-se uma chamada opcional; **Transmissão de arquivos apenas para APE:** - A chamada pode ser obrigatória (ou mandatória) no envio de PDU - No recebimento do PDU torna-se uma chamada opcional; **Transmissão e recepção de arquivos para APE:** - Pode ser obrigatória esta chamada no envio de PDU; - No caso do recebimento do PDU torna-se uma chamada opcional;
file-OfferPDU (File-OfferPDU)	**Recepção de arquivos apenas para APE:** - Esta chamada pode não ser exigida no envio de PDU - Em se tratando do recebimento do PDU, é uma chamada obrigatória; **Transmissão de arquivos apenas para APE:** - Esta chamada pode ser obrigatória (ou mandatória) no envio de PDU; - No caso do recebimento do PDU também é uma chamada obrigatória; **Transmissão e recepção de arquivos para APE:** - Pode ser obrigatória no envio de PDU; - No caso do recebimento do PDU, também é uma chamada obrigatória;
file-AcceptPDU (File-AcceptPDU)	**Recepção de arquivos apenas para APE:** - Esta chamada pode ser obrigatória (ou mandatória) no Envio de PDU; - No caso do recebimento do PDU torna-se uma chamada não exigida; **Transmissão de arquivos apenas para APE:** - Esta chamada pode não ser exigida no envio de PDU; - Para recebimento do PDU, é uma chamada obrigatória; **Transmissão e recepção de arquivos para APE:** - Chamada pode ser obrigatória no envio de PDU; - Para recebimento do PDU, também é uma chamada obrigatória;
file-StartPDU (File-StartPDU)	**Recepção de arquivos apenas para APE:** - Para o envio de PDU, esta chamada não é exigida; - No caso do recebimento do PDU, é uma chamada obrigatória; **Transmissão de arquivos apenas para APE:** - Esta chamada pode ser obrigatória no envio de PDU; - No recebimento do PDU, é uma chamada não exigida; **Transmissão e recepção de arquivos para APE:** - Chamada pode ser obrigatória no envio de PDU; - No caso do recebimento do PDU também é uma chamada obrigatória;

Mbft-NonStandardPDU (MBFT-NonStandardPDU)	**Recepção de arquivos apenas para APE:** - Esta chamada é opcional no envio de PDU ; - No caso do recebimento do PDU, também é uma chamada opcional; **Transmissão de arquivos apenas para APE:** - Esta chamada é opcional no envio de PDU - Para o recebimento do PDU, também é uma opcional; **Transmissão e recepção de arquivos para APE:** - Esta chamada é opcional no envio de PDU; - No caso do recebimento do PDU, também é uma opcional;

Em atividades de grupo como reuniões, conferências, entre outros, onde os participantes estão fisicamente dispersos, há a necessidade de troca de dados entre estes participantes. O termo comunicação multiponto descreve a interconexão de vários terminais. No T.127, o MBFT permite que arquivos possam ser trocados interativamente entre os participantes em um ambiente multiponto através de uma rede, independente da tecnologia subjacente, pelo Serviço de Comunicação Multiponto (MCS).

No método Private-Channel-Join-InvitePDU**,** podemos ter primitivas responsáveis pela identificação do canal, identificação dos dados do canal ou até mesmo o modo (por exemplo broadcast ou ponto a ponto). Acredita-se também, que ao limitar o número de chamadas deste método, pode-se também diminuir a complexibilidade da implementação, pois estamos limitando em uma conferência o número de canais para uma transferência de arquivo e com isso, menos dados de controle do canal, identificação do mesmo entre outros. A seguir, é mostrado a chamada Private-Channel-Join-InvitePDU, que foi encontrada na captura do tráfego proveniente de uma conferência com transferência de arquivo entre duas estações e um MCU (Figura 7). O pacote a ser apresentado, teve alguns códigos abstraídos para limitarmos os comentário somente ao T.120.

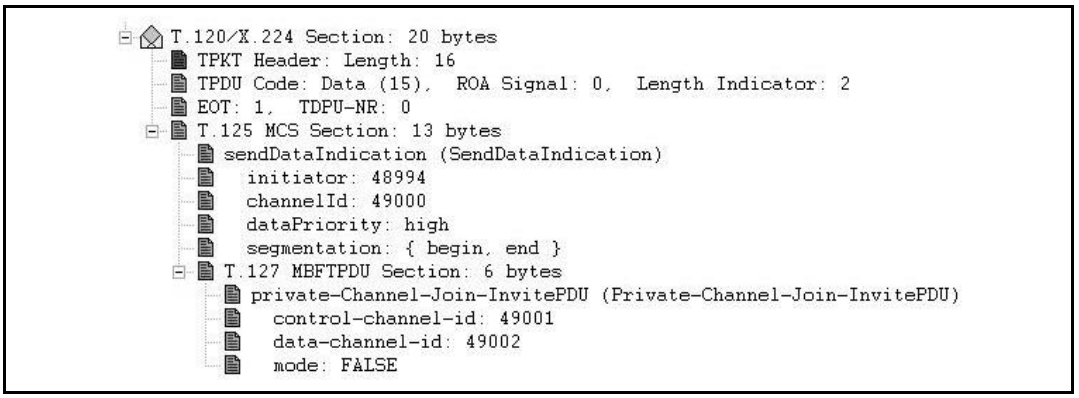


Figura 7: Pacote capturado contendo a chamada Private-Channel-Join-InvitePDU

O método Private-Channel-Join-ResponsePDU trata-se de uma resposta ao método anterior (Private-Channel-Join-InvitePDU), e conseqüentemente, está diretamente relacionado as primitivas de identificação de canal, identificação dos dados do canal, o modo que os dados trafegam no canal (broadcast ou ponto a ponto). Com isso, qualquer "poda" que possa a ser feita no método Private-Channel-Join-InvitePDU a mesma idéia pode ser seguida para o Private-Channel-Join-ResponsePDU.

No que se refere a chamada File-OfferPDU, ela pode ser "podada" no que diz respeito a recepção de arquivos apenas para APE, ou seja, essa não é exigida no envio de PDU para a APE. Uns dos motivos seria que a APE é a parte da aplicação de transferência de arquivo que envia aspectos que não têm nenhum efeito direto na interconexão (por exemplo, interface de usuário), podendo ser implementada em qualquer plataforma. Por ser executada na máquina cliente, a recomendação T-127 não cria restrições para a implementação. Uma aplicação de usuário confia

nos serviços de uma APE se comunicar com pontos de aplicações em outros nodos. Não há comunicação com MCS ou GCC; isto é feito pelo File APE. A aplicação de usuário inicia uma sessão de transferência de arquivo por seu File APE e especifica as capacidades de aplicação e modo de sessão. Uma vez estabelecida a sessão, todo as transações específicas do MBFT são executadas pelo File APE em nome da aplicação de usuário. Na figura 8, temos o pacote capturado com a chamada File-OfferPDU, os mesmos comentários sobre as abstrações da figura anterior, também servem para a seguinte.

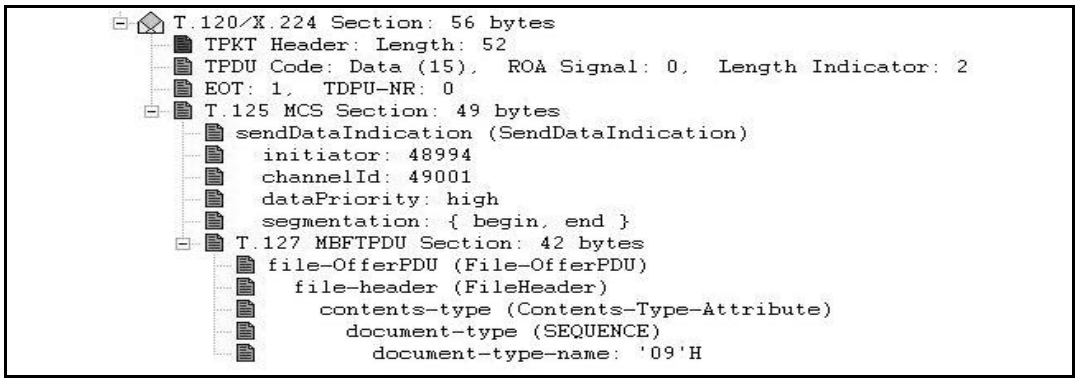


Figura 8: Pacote capturado contendo File-OfferPDU

No método File-AcceptPDU, não é exigida a implementação na recepção de arquivos para a APE quando relacionado ao recebimento de uma PDU, ou seja o aplicativo não é obrigado a receber um arquivo, ele pode simplesmente descartá-lo sem realizar nenhum tratamento. Em se tratando de envio de PDU para APE, este método também não é obrigatório, visto que o recebimento também segue o mesmo padrão. As mesmas considerações feitas no parágrafo anterior também servem para este, que a APE não têm efeito "direto" na interconexão da rede e portanto livre de plataforma de implementação. Figura 9, mostra pacote capturado com abstrações.

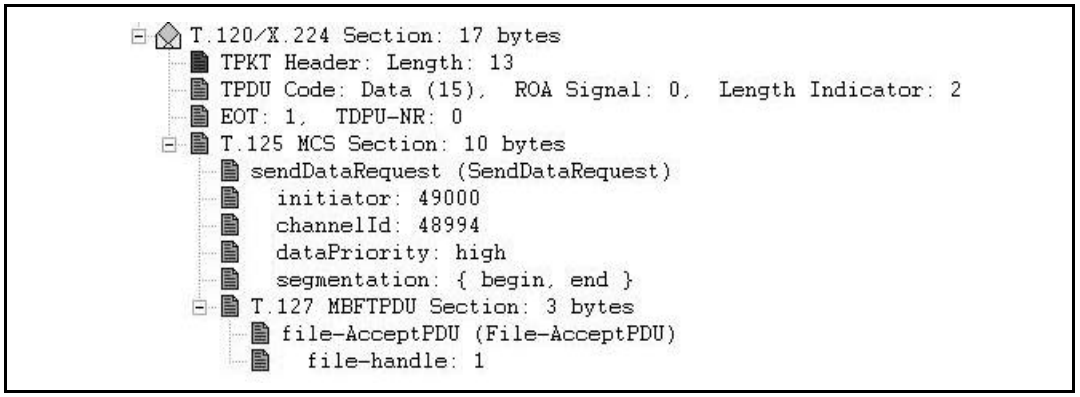


Figura 9: Pacote capturado contendo File-AcceptPDU

Para o método File-StartPDU, quando a chamada for para o envio de PDU (recepção de arquivo apenas para APE) esta não é exigida a implementação. A idéia é a mesma para o recebimento de PDU (na questão de transmissão de arquivo apenas para a APE). Essa "poda" é possível por não haver uma comunicação direta com MCS ou GCC, essa comunicação é realizada pelo File APE, voltando novamente ao termo sobre independência de plataforma.

O MBFT-NonStandardPDU permite transmitir qualquer informação não-padrão e permite a aplicações implementarem operações não-padrões. Uso de operações não-padrões podem estar

sujeitas à negociação de capacidades de não-padrão associadas. MBFT-NonStandardPDUs pode ser enviado a qualquer hora, mas em modo de administração só pode ser enviado por File APEs que foram concedidos o Privilégio de Não-padrão. É opcional este método segundo a especificação T.127, portanto não é necessário implementá-lo na: recepção de arquivos apenas para APE (envio e recebimento), transmissão de arquivos apenas para APE (envio e recebimento) e transmissão e recepção de arquivo para APE (envio e recebimento).

7 Considerações e Trabalhos Futuros

A colaboração visual ou mais especificamente a videoconferência é um termo cada vez mais difundido tanto no meio empresarial como no acadêmico. A grande causa é o fato de propiciar que as pessoas possam interagir e trabalhar colaborativamente em conjunto, mesmo que elas estejam geograficamente dispersas.

Com a evolução e aumento da colaboração visual, torna-se necessário um maior desenvolvimento e aceitação de protocolos padronizados para que serviços auxiliares de comunicação de dados possam acontecer. Baseado nisso o protocolo T.120 da União Internacional de Telecomunicações (ITU-T), vem ganhando espaço. Dentro desse contexto, o trabalho visou propor a criação de uma proposta minimalista que contenha as funções mínimas para o funcionamento de uma aplicação de transferência de arquivos sobre o T.120 com vistas a diminuir a complexibilidade na implementação das chamadas e primitivas definidas na recomendação. Com a análise do tráfego de uma videoconferência, pode-se constatar que existem algumas chamadas ou primitivas, que poderiam ser reduzidas ou eliminadas sem afetar a proposta da pilha T.120. Isto permitiria uma aplicação T.127 que seria executada com uma menor complexibilidade na implementação e maior eficiência na utilização da capacidade da rede, além de não perder algumas as características desejáveis tais como: a escolha de uma solução padronizada, que não limite as suas opções a produtos desenvolvidos por somente um fabricante, e que permita a comunicação através da Internet.

Como trabalhos futuros, poderiam ser desenvolvidas as demais séries do T.120, como o compartilhamento de aplicações, o serviço de quadro branco e também chat, e buscar a integração destas soluções em somente uma solução T.120 leve.

Referências

ITU-A. **ITU-T Recommendation H.323** – Packet-based multimedia communications systems. International Telecommunication Union: 125p, 1998.

ITU-B. **ITU-T Recommendation T.120** – Data protocols for multimedia conferencing. International Telecommunication Union: 25p, 1996.

ITU-C. **ITU-T Recommendation T.127 –** multipoint binary file transfer protocol. International Telecommunication Union: 63p, 1995.

ITU-D. **ITU-T Recommendation T.120 – Annex C** Data protocols for multimedia conferencing - Lightweight profiles for the T.120 architecture. International Telecommunication Union: 63p, 1995.

LEOPOLDINO, G. Modelos de Comunicação para Videoconferência, Março, 2002, Disponível em: <http://www.rnp.br/newsgen/0105/video.shtml>

LEOPOLDINO, G. e MOREIRA, E. **Avaliação de Sistemas de Videoconferência.** Março, 2002, Disponível em: <http://www.rnp.br/arquivos/AvaliacaoVideo.pdf>

PINHEIRO, C. **Tutorial – Especificação ITU H.323**, Março, 2002, Disponível em: <http://penta2.ufrgs.br/h323/indroducao.html>.

Videoconferencing Cookbook Version 3.0. Dezembro, 2002, Dipsonível em: <http://www.videnet.gatech.edu/ cookbook>.

Modelagem e Análise de Desempenho de Uma Rede Baseada em Tecnologia MPLS

Aujor Tadeu Cavalca Andrade
Tadeu@lrg.ufsc.br

Carlos Becker Westphall
Westphall@lrg.ufsc.br

Laboratório de Redes e Gerência – LRG
Centro Tecnologia – CTC
Universidade Federal de Santa Catarina – UFSC

Resumo. Este artigo apresenta uma avaliação comparativa entre a tecnologia atual de comutação IP e o emprego da mesma com MPLS. O propósito é analisar e avaliar o desempenho do MPLS e verificar suas principais métricas de transmissão, através da simulado de cenários. Embora questões de implementação e infraestrutura sejam consideradas, esta avaliação visa contribuir para o desenvolvimento e aperfeiçoamento das tecnologias de comutação**.**

Palavras-chaves: MPLS, transmissão de dados, análise de desempenho**.**

1. Introdução

Com o avanço tecnológico e o uso crescente das redes de computadores, novas tecnologias são necessárias para atender os requisitos pelo aumento de transmissão de dados, processamento de alta performance e qualidade de serviço, estimulando o surgimento de novas tecnologias e o aperfeiçoamento das já existentes.

O desenvolvimento de uma tecnologia de transmissão de dados que permita transportar grandes quantidades de dados em altas velocidades a grandes distâncias, possibilitando uma otimização e funcionalidade, é de extrema importância.

A maior parte das propostas neste sentido concentra-se no aperfeiçoamento do processo de comutação de pacotes, o que possibilita um considerável ganho de desempenho na transmissão dos dados como um todo.

O propósito é analisar uma destas tecnologias: o MPLS – *MultiProtocol Label Switching*, uma tecnologia que permite ampliar o desempenho de tecnologias de redes já existentes. O MPLS representa o próximo passo da evolução baseada em padrões, combinando tecnologias de comutação da camada 2 (camada de enlace) com as tecnologias de roteamento da camada 3 (camada de rede).

Este artigo permite uma comparação entre os métodos atuais de roteamento e a utilização desta tecnologia. Para esta comparação será utilizado um software de modelagem e simulação, o *network simulator* 2, com o intuito de analisar aspectos da transmissão de dados e conjeturar sobre o encaminhamento de pacotes baseado na tecnologia MPLS. O artigo é apresentado na seguinte forma. Na seção 2, descrevemos os componentes MPLS e suas funcionalidade. Na seção 3 temos a arquitetura MPLS. Na seção 4 descrevemos a modelagem e a simulação do cenário. A seção 5 apresenta a inferência sobre os resultados e trabalhos futuros.

2. MultiProtocol label switching – MPLS

2.1 LSR – Label switching routers, LSP – label switch path

Roteadores de comutação por rótulos (LSR) são equipamentos que realizam a comutação no protocolo MPLS. Um LSR é um dispositivo que aumenta a velocidade de encaminhamento no núcleo de uma rede MPLS. Quando um LSR localiza-se na periferia da rede MPLS denomina-se LSR de borda *(Edge LSR)*, enquanto que aqueles situados no núcleo da rede denominam-se LSR de núcleo *(Core LSR)*. Ao conjunto de LSR denomina-se nuvem MPLS *(MPLS Cloud)*.

Os LSR participam no estabelecimento do LSP (label switch path) caminhos comutados por rótulos. Os LSPs são determinados por ação do protocolo do plano de controle ou por ação de gerência de rede. As determinações das rotas para um LSP podem ser definidas com auxílio de protocolos de roteamento convencional. Os LSPs são unidirecionais, isto é, suportam encaminhamentos de datagramas em um único sentido de modo que um LSP pode definir uma seqüência ordenada de LSRs. Ao primeiro LSR chamamos de LSR ingresso (Ingress LSR) e o último de LSR egresso (Egress LSR).

No estabelecimento de uma sessão para distribuição de rótulos necessitamos sempre de LSR pares (LSR par: é o conjunto de LSR adjacentes, onde é estabelecida uma sessão para fins de distribuição de rótulos.). Dependendo do sentido do fluxo de datagramas no LSP, o LSR a jusante (primeiro equipamento que faz parte do LSR par, responsável por selecionar o rótulo que será utilizado), seleciona o rótulo que o LSR a montante (LSR a montante: segundo equipamento que faz parte do LSR par utiliza os rótulos que foram selecionados efetuar o encaminhamento), deve utilizar no encaminhamento.

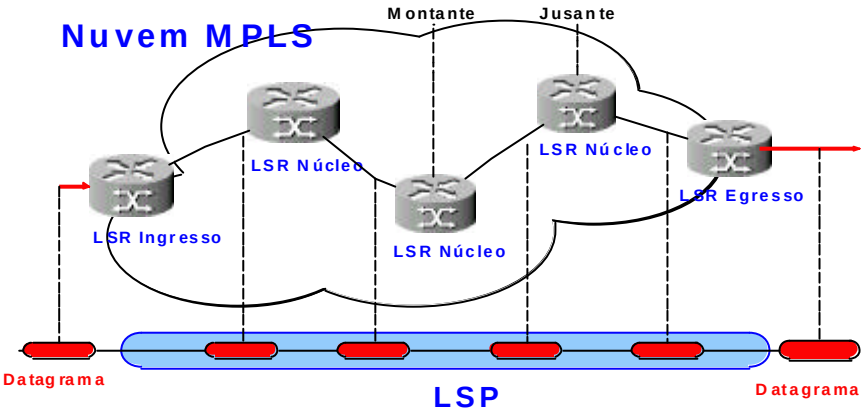


Figura 1 - Principais elementos MPLS.

2.2 Classe de equivalência para remessa – forward equivalence label – FEC

Uma FEC representa condições determinadas para verificar se um dado datagrama pertence ou não a FEC estabelecida e se compartilham das mesmas exigências para seu transporte.

Em todo o pacote a um grupo específico e a este é dado o mesmo tratamento da origem ao

destino, ao contrário da remessa convencional do IP, em MPLS, a atribuições de um determinado pacote a uma FEC são baseadas em exigências de serviços ou simplesmente no prefixo do endereço.

Cada LSR constrói uma tabela para especificar como um pacote deve ser enviado. Existem dois tipos de elementos FEC:

- Prefixo de rede: no caso IPv4, possui comprimento arbitrário de 0 a 32 *bits.*
- Endereço de nó: é um endereço IP de classe A, B ou C.

Um elemento FEC pode contemplar ainda informações adicionais sobre:

- A origem do datagrama (endereço IP de origem);
- A carga do datagrama (protocolo e *ports* de transporte);
- Parâmetros de qualidade de serviços tais como:
- Precedência do quadro, conforme definido pelo padrão IEEE 802.1d; classe de serviço, conforme definida pela arquitetura de serviços diferenciados *(DiffServ).*
- Precedência do datagrama, conforme definida pelo campo TOS (*Type of Servece)* do protocolo IP.

2.3 Rótulos e ligação de rótulos

Um rótulo, em sua forma mais simples é atribuído na camada de enlace e identifica o trajeto que um pacote deve seguir. O roteador de recepção examina o pacote e verifica o índice do rótulo para determinar o próximo *hop*.

Uma vez o pacote rotulado, o encaminhamento deste para o destinatário este baseado na troca do rótulo, isto é, cada valor do rótulo é de significado apenas local pertencendo somente aos pulos entre LSRs. Quando um pacote for classificado com uma FEC novo ou existente, um rótulo será distribuído ao pacote. Os valores do rótulo são responsáveis pelo envio dos pacotes e são baseados na camada de dados.

O rótulo é encaixado entre a camada de dados e camada de rede. Atribuição do rótulo permite uma melhor qualidade de serviço de remessa nos seguintes casos:

- distribuição *unicast*;
- engenharia de tráfego;
- *multicast*;
- rede confidencial virtual (VPN);

2.4 Criação do rótulo

Há diversos métodos usados na criação dos rótulos:

- Método baseado na topologia – usa protocolo de distribuição normal como o OSPF (Abrir primeiro menor caminho) e o BGP (Protocolo gateway de borda).
- Método baseado na requisição – usa processo baseado nas requisições de fluxo de controle de dados como o RSVP (Protocolo de reserva de recursos).

Método baseado no tráfego – usa a recepção do pacote para produzir a atribuição e distribuição da etiqueta.

2.5 Trajetos das trocas de rótulo

Uma coleção de dispositivos MPLS forma um domínio MPLS. Dentro deste domínio, o trajeto das taxas de rótulos é determinado para um pacote de dados juntamente com uma FEC,

estabelecendo se o pacote pertence a um grupo específico. O componente responsável por determinar a distribuição de trajetos antes da transmissão é o LSP. Quanto à forma de distribuir o LSP possui:

- Distribuição do *hop-by-hop*: Cada LSR seleciona independentemente do próximo *hop* do pacote aqueles pertencentes a uma FEC determinada. Este metodologia é similar às usadas em redes IP como BGP e a OSPF.
- Distribuição explícita: Nesta distribuição o LSR de ingresso determina a lista dos nós em que o pacote irá passar. Seu trajeto determinado pode assegurar qualidade no serviço e no tráfego dos dados, possibilitando uma otimização do fluxo de toda a rede. O estabelecimento do LSP para uma FEC é unidirecional o seu retorno deve fazer exame de outro LSP.

2.6 União do rótulo

Os fluxos de pacotes de diferentes interfaces podem ser rotulados e comutadas juntos usando um rótulo comum, como se estiverem atravessando a rede para um mesmo destino. Isto é conhecido com agregação dos fluxos.

2.7 Retenção do rótulo

O MPLS define o tratamento para as ligações dos rótulos recebidos do LSRs, que não são os próximos pulos determinados pela FEC.

São dadas duas definições:

- Método Conservativo – Neste método, a ligação entre o rótulo e uma FEC recebida pelo LSR, passa por uma verificação buscando identificar se este LSR pertence à próxima seqüência do pulo, caso não pertença o pacote é rejeitado.
- Método liberal – Nesta modalidade, as ligações entre o rótulo e uma FEC recebida do LSR, passa pela mesma verificação do método conservador, porém caso o resultado seja negativo o pacote ainda é aceito. Esta modalidade permite uma adaptação rápida às mudanças de topologia e auxilia na distribuição do tráfego ao outro LSP.

2.8 Controle de rótulos

O MPLS define os seguintes métodos para distribuição dos rótulos entre LSRs vizinhos:

- Método independente – neste método, um LSR reconhece uma FEC particular e não pertencente ao seu grupo especifico e faz a decisão de encaminhar o pacote independentemente da FEC do próximo pulo.
- Método ordenado – neste método, um LSR só reconhece uma FEC pertencente ao seu grupo especifico, caso apareça um pacote não identificado o mesmo não é repassado.

3. Arquitetura MPLS

Aumentar a velocidade e eficiência de redes IP é essencial para um desenvolvimento de novas aplicações. O encaminhamento tradicional IP é fonte de estudos, pois gera problemas de crescimento que são :

- O atraso na transmissão do datagrama;

- Congestionamento de redes;

A comutação IP visa minimizar os problemas de atraso de datagramas no nível de enlace e engenharia de tráfego busca anular o congestionamento da rede. Neste contexto, o propósito do MPLS é melhorar o desempenho do roteamento de datagramas e evitar congestionamentos na rede.

Inicialmente, as tecnologias de comutação utilizavam hardware ATM para comutação de IP por rótulos e eram soluções aplicadas nos protocolos de enlace e redes. Essas tecnologias, na metade dos anos noventa, sofreram mudanças em sua utilização, continuaram com hardware ATM, por sua velocidade e incorporaram sinalização IP pensando em roteamentos integrados com endereços IP.

No modelo de rede MPLS, quando um datagrama chega ao elemento de entrada da rede denominado LSR ingresso, este busca classificá-lo em uma FEC específica, para a qual já exista um LSP definido. Seu LSP tem sentido de fluxo unidirecional e define uma seqüência ordenada de LSRs onde o último é chamado de LSR egresso.

O rótulo deste datagrama gerado por um LSP tem seu sentido quando o LSR a jusante seleciona o rótulo do LSR a montante. Neste caso, um par de LSR se forma e são chamados de LSR par e estabelecem uma sessão para distribuição de rótulos.

A arquitetura MPLS tem duas opções para estabelecimento de rotas, onde o método escolhido será utilizado durante o estabelecimento de um LSP:

- Roteamento *hop-by-hop;*
- Roteamento explícito;

No roteamento *hop-by-hop*, há uma forma independente de escolha de cada nó para um próximo hop associado a FEC, este tipo de escolha é a forma tradicional do roteamento IP. Um LSP roteado hop-by-hop é um LSP cuja rota é selecionada pelo roteamento hop-by-hop. No roteamento explícito, o LSP roteado não são autônomos para escolher o próximo *hop*. Assim, um único par LSR estabelece todas as partes das rotas, o primeiro LSR ingresso e o último LSR egresso que compõem a rota LSP. Temos ainda casos que se a especificação de todos os LSRs da rota, então chamamos LSP roteado explicitamente de forma estrita. Caso sejam algumas partes do LRS, então denominamos de LSP roteado explicitamente de forma fraca.

A utilização de rótulos em redes orientada a conexão são úteis para identificar conexões e associações de pacotes. O MPLS permite o estabelecimento de caminhos comutados (LSPs), utilizando rótulos para identificá-los.

O LSR ingresso utiliza esta associação para decidir qual o encaminhamento dar ao datagrama quando inserido nesta rede MPLS. Quando o datagrama pertence ao uma determinada FEC, LRS ingresso encaminha o datagrama através da comutação por rótulo LSP, caso contrário o encaminhamento IP padrão *hop-by-hop*.

Entre as associações feitas pela FEC temos duas perpectivas;

- Quando do estabelecimento de um LSP, para se identificar através da FEC o próximo *hop* da rota;
- Quando as tabelas de roteamento são atualizadas, para certificar-se através da FEC se o próximo *hop* da rota estabelecida anteriormente para o LSP, ainda permanece ao mesmo.

Após as associações feitas pela FEC ao LSP, a sequência realizada por um LSR de egresso é:

- A determinação do endereço *host* pela FEC que coincide com o endereço do LSR ou de um *host* diretamente conectado;
- A FEC especifica um prefixo de rede correspondente a uma das interfaces do LSR;

- O próximo *hop* da rota é um roteador fora do domínio MPLS;

A eficiência na utilização de rótulos está ligada ao seu posicionamento no quadro de enlace, para permitir a comutação de pacotes sem processamento na camada de rede. Os identificadores de conexão de enlace devem coincidir com os rótulos.

3.1 Formato do datagrama

Para tecnologias de enlace que não empregam identificadores de conexão o MPLS define uma rotulação denominada encapsulamento genérico. Esta estrutura chamada *shim header* armazena o rótulo ao datagrama de forma posicionada entre o cabeçalho de enlace e sua datagrama IP.

Esta estrutura é compostas por:

- Rótulo de 20 bits;
- Campo experimental EXP de 3 bits;
- Campo TTL *time to live de* 8 bits;

um campo B de 1 bit cujo objetivo é indicar se o rótulo corresponde, ou não, ao último de uma pilha de rótulos, o que permite o encapsulamento de múltiplos rótulos.

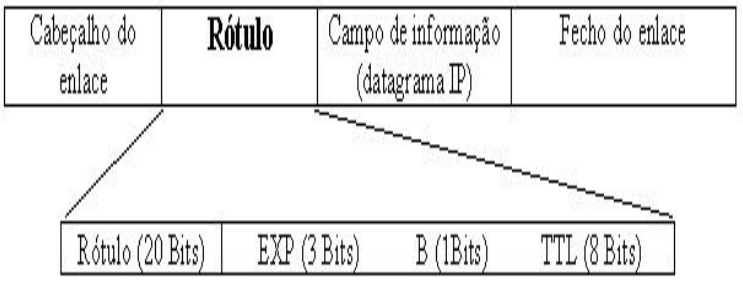


Figura 2 - Estrutura *Shim Header*

A arquitetura MPLS possibilita organização de vários rótulos utilizando-se do método *First in First out* (FIFO). O ponto fundamental de envolvimento de pilhas de rótulos é o encaminhamento realizado nos LRSs, onde é sempre baseado no rótulo encontrados no topo da pilha. Esta possibilidade permite definirmos uma hierarquia de rótulos na pilha e implica em importantes aspectos de roteamento e tunelamento de informações.

Antes de tratar de distribuição dos rótulos, é preciso deixar alguns pontos bem claros como:

- Os rótulos são distribuídos entre LSR quando do estabelecimento de LSP;
- O rótulo está sempre associado a uma FEC;

Os rótulos prevêem formas de distribuição onde se especifica que os dados têm um sentido de fluxo e estão sempre associados a uma FEC, onde qualquer associação é criada a jusante *(downstream)* e distribuído à montante *(upstream)*. Então qualquer associação criada por um LRSs são distribuídas somente por seus pares a montante.

Cada LRS contém, conforme a figura 3, uma base de informações de rótulos ou LIB (*label information base)* a qual tem:

Figura 3 – Distribuição de Rótulos.

- FEC;
- Rótulo de entrada (distribuição para LSR a montante);
- Interface de entrada
- Rótulo de saída (recebido do LRS à jusante);
- Interface de saída;

4. Modelagem e simulação de uma rede MPLS

A fim de demonstrar as características e capacidade da tecnologia MPLS citadas anteriormente, foi modelada uma rede com estas características. Em seguida, esta rede foi alvo de simulações que permitiram demonstrar o seu funcionamento.

Este secção apresenta o processo de modelagem e simulação, além de apresentar os resultados obtidos a partir das iterações realizadas.

4.1. Network simulator 2 (NS2)

Para a modelagem da rede e o processo de simulação, foi utilizada a ferramenta *Network Simulator* 2, ou simplesmente NS2. O NS2 é uma ferramenta que tem o propósito de executar modelagens e simulações de situações reais, com recursos de capturar detalhes e possibilitam inúmeros cenários e condições para simulação. As principais características do *network simulator* são:

- Possibilidade de introduzir acontecimentos e condições, de forma a criar um ambiente de condições iguais as reais;
- Diversidade de molduras, esqueletos, cenários e arquitetura de redes;
- Simulador de Multi protocolos;
- Tem opção de orientação a objeto;

4.2. Modelo da rede MPLS

O modelo de rede MPLS utilizado neste trabalho foi modelada para ser uma rede simples com poucos nós, sendo que alguns destes formando uma “nuvem MPLS”.

Como mostrado na figura 4, esta rede é composta de 11 nós, numerados de 0 a 10. Os nós 0, 1, 9 e 10 foram modelados como nós comuns, não-MPLS, enquanto que os demais foram definidos como nós MPLS. O nó 2, está posicionado na rede como um LSR ingresso e o nó 8 como um LSR egresso. Isto se deve ao fato de todo o tráfego definido na rede originar-se dos nós 0 e 1; e terem como destino os nós 9 e 10.

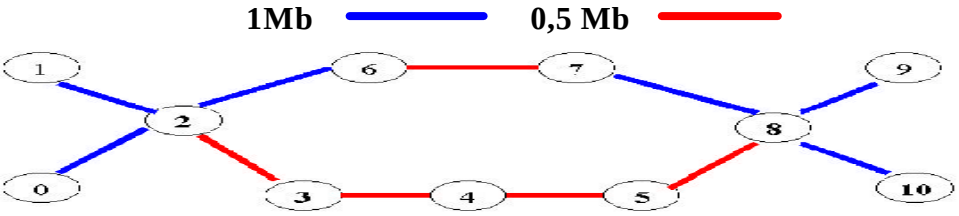


Figura 4 – Disposição nós na Rede

Os *links* de dados entre os elementos da rede foram modelados como duplex com atraso de transferência de 10 ms e possuem capacidades de 0.5 ou 1.0 Mbps.

4.3 Medidas de desempenho

A ferramenta de animação “nam” foi utilizada para visualizar a simulação. Através desta pode-se observar de maneira clara o comportamento da rede ao longo de toda a simulação.

Entretanto, esta ferramenta não extrai dados que permitam analisar o desempenho da rede. Assim, foram estabelecidas três medidas de desempenho a serem analisadas a partir das iterações: o troughput, o atraso dos pacotes e a perda de pacotes. Para isso, foram implementados alguns *scripts perl* para filtrar o arquivo. de *trace*, criado pelo NS2 a cada iteração, e apresentar estas medidas.

4.4. A Simulação

Para ilustrar o impacto do uso da tecnologia MPLS, optou-se por organizar a simulação em dois momentos: num primeiro momento a rede deverá funcionar como uma rede IP tradicional, utilizando-se do processo de encaminhamento tradicional, num momento seguinte, esta mesma rede deve se beneficiar das características da tecnologia MPLS através da definição de um LSP entre os nós 2 e 8.

Com isso, pretende-se comprovar que após o estabelecimento deste LSP, os *links* de enlace passem a ter uma vazão constante, o atraso dos pacotes diminua e a perda de pacotes seja reduzida a uma quantidade mínima.

A figura 5 mostra o resultado da vazão (*throughput*) em Kbit/s para a simulação da rede modelada. Pode-se observar a partir do gráfico apresentado, que a vazão se estabiliza logo após a criação do LSP entre os nós 2 e 8. Isto deve-se ao fato de que após a criação deste LSP, os dados tendem a seguir de forma mais rápida, pois não se acumulam por muito tempo na fila, o que no fim acaba por permitir que sejam transmitidos mais bits num mesmo espaço de tempo quando comparado com o encaminhamento IP tradicional.

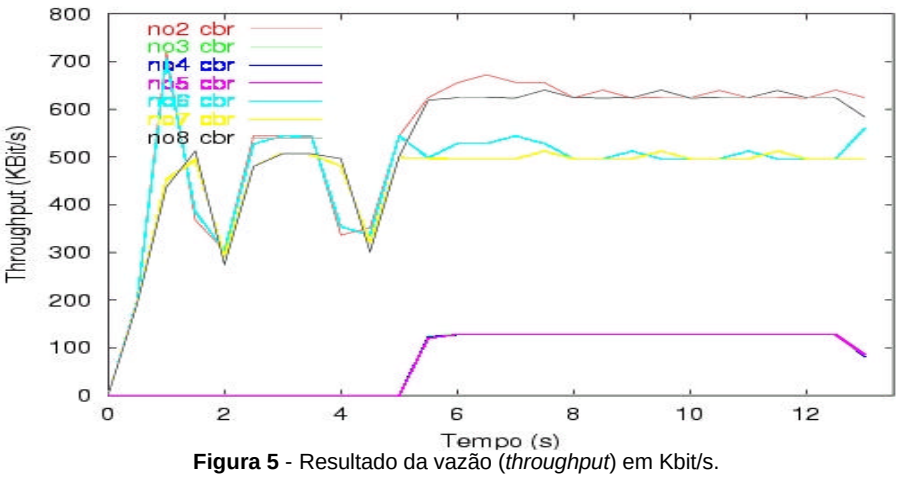


Figura 5 - Resultado da vazão (*throughput*) em Kbit/s.

No quadro 1 temos a relação tempo e ação, demonstrando detalhadamente as ações efetuadas em seu instante de tempo.

Tempo (s)	Ação
0.00	Inicio da Simulação.
0.05	Inicio do tráfego do nó 0.
0.10	Inicio do tráfego do nó 1.
4.80	Configura um LSP entre o nós 2, 8, passando pelos nós 3, 4, e 5.
5.00	Ativa o LSP no nó 2.
13.00	Encerra as fontes de tráfego.
13.50	Encerra a simulação

Quadro 1 – Relações de ação e tempo.

O aumento da vazão está associado à diminuição do atraso dos pacotes, como pode-se observar a partir da figura 6. Por agregar as funções de *switching* e *forward* na camada de enlace, o MPLS encaminha os pacotes de forma mais rápida. Além disso, pode-se notar que o MPLS acaba por controlar este atraso, tornando-o previsível para determinados tráfegos, qualidade que é desejável para a QoS (*Quality of Service*).

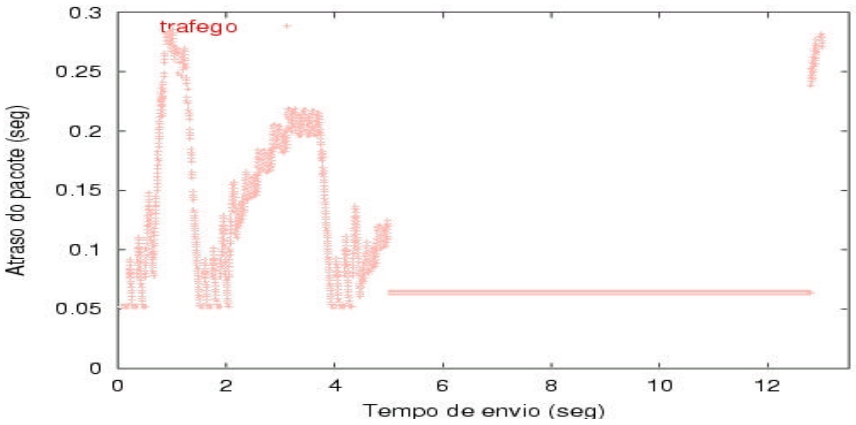


Figura 6 – Relação entre atraso de pacotes e tempo de envio.

Para complementar a análise do desempenho da rede operando com as características MPLS, foi medida a taxa de perda de pacotes ao durante a simulação. No primeiro momento da simulação, sem as características MPLS, pôde-se constatar uma taxa de perda de aproximadamente 7,75%, enquanto que após a introdução do LSP esta taxa caiu para aproximadamente0,98%. Este resultado é perfeitamente compreensível, pois ao agilizar o encaminhamento dos pacotes o MPLS diminui a possibilidade de descarte destes, diminuindo portanto a taxa de perda destes pacotes.

5. Conclusão

Quando foi iniciado o estudo da tecnologia MPLS, o propósito era de comprovar se as melhorias do processo de comutação de pacotes e da capacidade de roteamento eram realmente eficazes e eficientes.

A partir das simulações realizadas comparando uma rede baseada no roteamento convencional e uma rede com conceitos MPLS, percebemos que o comportamento do processo de comutação de pacotes era totalmente diferente.

A rede MPLS após o estabelecimento do LSP tornava, o *link* de enlace com vazão constante, implicando na diminuição do atraso de pacotes e na perda do mesmo. Esta constatação nos confirma uma das características do emprego desta tecnologia que oportuniza a melhora no desempenho de roteamento e também flexibilidade na introdução de novos serviços.

O beneficio desta característica vai de encontro com uma dos problemas atuais da rede de computadores que é o atraso na propagação, devido ao roteamento atual ser baseado nas rotas mais curtas. Uma vez criado o LSP, a uma maior transmissão dos dados em um mesmo espaço de tempo, possibilitando uma otimização em sua funcionalidade.

Outro detalhe de fundamental importância constatado nos gráficos gerados e no funcionamento desta tecnologia é o fato da camada de enlace agregar funções de *switching* e *forward*. Estas qualidades permitem encaminhar pacotes rapidamente e controlar o atraso no tráfego.

Uma dos dados mais significativos gerados foi o percentual da perda de pacotes utilizando a tecnologia MPLS. Nesta simulação, na rede tradicional a taxa de perda de pacotes é aproximadamente 6 vezes maior em comparação com a rede MPLS. Este dado demonstra outra característica do MPLS que é a eficiência no encaminhamento e redução dos descartes de pacotes.

A pesquisa da tecnologia MPLS visa contribuir para a melhoria da qualidade de serviço utilizada nas redes atuais, por ser uma tecnologia de aplicabilidade real possibilita a comunidade desfrutar de suas funcionalidades e benefícios.

Em trabalhos futuros, novas simulações mais detalhadas com vários cenários serão realizadas, estas condições serão necessárias para possibilitar uma maior confiabilidade no uso da tecnologia MPLS.

6. Referências Bibliográficas

[1] CHIOZZOTTO, M. & SILVA, L. **TCP/IP – Tecnologia e Implementação**. São Paulo, Erica, 1999.

[2] COMER, Douglas E. **Interligação em redes com TCP/IP: Princípios, Protocolos e Arquitetura**. Rio de Janeiro: Campus, 1998.b

[3] Davis, Bruce. ***MPLS Technology and Applications***. São Paulo: Ed. da USP, 1979.

[4] DOWES, Kevin; FORD, Merille; LEW, H. KIM. **Internet Working Techinogies Handbook**. Ed 2ª. Tradução de Fabio Freitas. Rio de Janeiro: Campus, 2000.

[5] MAGALHÃES, M. & CARDOZO, E. **Introdução à comutação internet protocol por rótulos através de multiprotocol label switching**. In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES, 25, 2001 Florianópolis. Resumos... Florianópolis: Simpósio Brasileiro de Redes Computadores, 2001, p. 1-48.

[6] NETWORKSIMULATOR. Ns by examples. **Aplicações, característica da ferramenta e exemplos de simulações**. Disponível em: <http://www.isi.edu/nsnam/ns. Acesso em: 10 set. 2001.

[7]SIMPOSIO BRASILEIRO DE REDES DE COMPUTADORES, 4., 2001, Florianópolis. Anais eletrônicos... Florianópolis: UFSC, 2001. Disponível em: <http://www.sbrc.br/anais/anais.htm>. Acesso em: 23 mar. 2001.

[8] SOARES, Fernando Gomes; LEMOS, Guido; COLCHER, Sérgio. **Redes de computadores das Lans, Mas e Wans às redes ATM**. Rio de Janeiro: Campus, 1997.

[9] SUAVE, J. P.; TEIXEIRA, J. H. Jr.; MOURA, J. **Redes de Computadores – Serviços Administração e Segurança**. São Paulo: Makron Books, 1999.

[10] TANEMBAUM, Andrew S. **Redes de Computadores**. Tradução de terceira edição. Rio de Janeiro: Campus, 1997.

[11] WEBPROFORUMS. Web ProForums. **Apresentação da tecnologia multiprotocol label switching e suas características**. Disponível em: <http://www.iec.org/tutorials/mpls. Acesso em: 04 mar. 2001.

Índice de Autores

Adriano Gheller Bruschi 9
Aidê Luzia A. de Souza 33
Alexander Roberto Valdameri 169
Arthur Tórgo Gómez 179
Aujor Tadeu Cavalca Andrade 207
Carla Gomes de Faria 71
Carlos Becker Westphal 121
Carlos Becker Westphall 207
Cleber Machado Ortiz 193
Cristiano Fornari Colpani 169
Cristiano Freese 141
Daniel Moser Trallamazza 23
Daniel Weller 9
Deborah Ribeiro Carvalho 131
Eduardo Erle dos Santos 121
Fernando Luiz Koch 121
Ismênia Ribeiro 85
Jaderson Trierweiler 23
Joyce Martins 23
Leandro Toss Hoffmann 179
Liane Margarida Rockenbach Tarouco 193
Lisandro Zambenedetti Granville 193
Luiz Ricardo Lopes 131
Marcel Hugo 57
Marcos Bueno 131
Marcos Dias Assuncao 121
Marcos José Santana 155
Maria Inés Cstiñeira 33
Mauricio Capobianco Lopes 45, 99
Miguel Alexandre Wisintainer 141
Omar Andres Carmona Cortes 155
Paulo César Rodacki Gomes 23
Rafael de Santiago 113
Regina Helena Carlucci Santana 155
Rogerio Sorroche 45, 99

Romeu Gadotti 57
Rosario Girardi 71, 85
Roselane M. M. Camargo 33
Rudimar Luís Scaranto Dazzi 113
Vera Schuhmacher Niedersberg Schumacher 33
Wilson Alves Neto 131